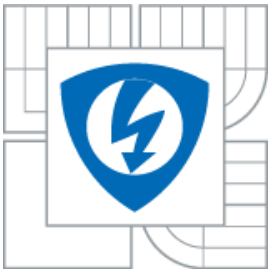
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

OCHRANA SOUKROMÍ NA INTERNETU

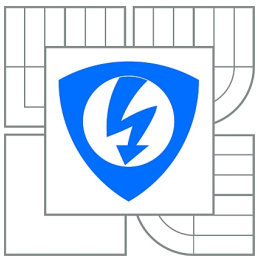
INTERNET PRIVACY PROTECTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. LUKÁŠ MALINA
AUTHOR

VEDOUCÍ PRÁCE — Ing. JAN HAJNÝ
SUPERVISOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Lukáš Malina	***ID:***	77735
Ročník:	2	***Akademický rok:***	2009/2010

NÁZEV TÉMATU:

Ochrana soukromí na Internetu

POKYNY PRO VYPRACOVÁNÍ:

Práce je zaměřena především na analýzu současných metod autentizace a související problém udržení soukromí uživatelů. Student se během diplomové práce zaměří především na teorii související se skupinovými podpisy, elektronickými mincemi a dalšími kryptografickými primitivy pro udržení soukromí uživatelů komunikačních systémů. Cílem práce je rozbor současných metod, jejich srovnání a zhodnocení jejich použitelnosti pro praktickou implementaci, jenž bude provedena na základě výběru vhodného protokolu v prostředí .NET.

DOPORUČENÁ LITERATURA:

[1] STALLINGS, William. Cryptography and Network Security. 4th edition. [s.l.] : [s.n.], 2006. 592 s. ISBN 0131873164.
[2] LIPMAA, Helger. Group signature schemes [online]. c2009 [cit. 2009-10-06]. Dostupný z WWW:<http://research.cyber.ee/~lipmaa/crypto/link/signature/group.php>.
[3] CAMENISCH, Jan, LYSYANSKAYA, Anna. A Signature Scheme with Efficient Protocols. [s.l.] : [s.n.], 2002. 21 s.

Termín zadání:	29.1.2010	***Termín odevzdání:***	26.5.2010

Vedoucí práce: Ing. Jan Hajný

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Anotace

Anonymní autentizace slouží k autentizaci uživatelů bez odhalení jejich vlastních identifikačních údajů či osobních dat. Technologie Anonymních Autentizačních Systémů (AAS) poskytuje ochranu soukromí uživatelů a zároveň zajišťuje bezpečnost systému. Tato práce představuje základní kryptografická primitiva, kterými se anonymní autentizace může zajišťovat. Mezi tato primitiva patří některé asymetrické kryptosystémy, avšak nezbytnou součástí tvoří například protokoly na bázi nulové znalosti, slepá podpisová schémata, prahová skupinová schémata, atd., která jsou představena v kapitole 1.

Obecně mají autentizační anonymní systémy uplatnění v aplikacích, jako jsou elektronické mince, elektronické hotovosti, skupinové elektronické podpisy, anonymní přístupové systémy, elektronické volby, atd., které jsou postupně analyzovány a představeny v kapitolách 2 a 3. V praktické části práce, která je popsána v kapitole 4, je představena implementace (v prostředí .NET v jazyce C#) systému AAS, který je vyvíjen na FEKT VUTBR.

Klíčová slova:

Autentizace, anonymní autentizace, soukromí, protokol nulové znalosti, skupinový podpis, elektronická mince, elektronická hotovost, anonymní autentizační systém.

Abstract

Anonymous authentication is a mean of authorizing a user without leakage of user personal information. The technology of Anonymous Authentication Systems (AAS) provides privacy of the user and yet preserves the security of the system. This thesis presents the basic cryptographic primitives, which can provide anonymous authentication. Among these primitives there are usually some asymmetric cryptosystems, but an essential part of anonymous authentication is based on zero knowledge protocols, blind signature schemes, threshold group schemes, etc., that are presented in Chapter 1.

Generally, Anonymous Authentication Systems have application as electronic coin, electronic cash, group signatures, anonymous access systems, electronic vote, etc., which are analyzed and presented in Chapters 2 and 3. In the practical section, the implementation (in the environment .NET in C#) of the AAS system is presented and described in Chapter 4, which is being developed at the FEEC BUT.

Key words:

Authentication, anonymous authentication, privacy, zero knowledge protocol, group signature, electronic coin, electronic cash, anonymous authentication system.

Bibliografická citace

MALINA, L. *Ochrana soukromí na Internetu.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. 75 s. Vedoucí diplomové práce Ing. Jan Hajný.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Ochrana soukromí na Internetu jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedeného diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne ....

podpis autora

Poděkování

Děkuji vedoucímu práce Ing. Janu Hajnému za velmi užitečnou metodickou pomoc a cenné rady při zpracování diplomové práce.

V Brně dne ....
podpis autora

OBSAH

ÚVOD

V dnešní době je stále větším trendem využívání Internetu k výměně informací, obchodním činnostem, ke komunikaci a k mnoha dalším aktivitám. Tyto činnosti usnadňují dennodenně život bezpočtu lidem, firmám a organizacím, kteří požadují funkčnost a bezpečnost výše zmíněných aktivit. Abychom zajistili potřebnou bezpečnost dat a informací, které putují skrz Internet, využíváme k tomuto účelu řadu kryptografických ochran. Z mnoha dobrých důvodů se k funkčnosti a bezpečnosti služeb a činností provozovaných na Internetu nyní přidává i ochrana soukromí či anonymita uživatele služeb. Mezi tyto aplikace a činnosti, kde je vhodné poskytovat ochranu soukromí, patří například elektronické hlasování (v ang. e-voting) nebo anonymní nakupování a pořizování aktiv pomocí elektronických mincí (v ang. e-cash). Jedná se tedy o snahu přenést běžné záležitosti, jako např. nakupování v obchodech, kde má osoba zaručenou anonymitu při platbě penězi, na ekvivalentní elektronické aplikace běžící na Internetu. Například použitím elektronické mince.

Práce je zaměřena na analýzu současných metod autentizace a související problém udržení soukromí uživatelů. Tyto metody obecně stojí na asymetrických kryptografických systémech a schématech, které jsou vhodně navrženy tak, aby zajistily autentičnost (původnost) dat, popřípadě entit či osob. Problémem, kterým se práce zabývá, je bezpečná autentizace entit a přitom zachování soukromí neboli anonymity uživatelů. Práce je založena především na teorii související se skupinovými podpisy, elektronickými mincemi a dalšími kryptografickými primitivy pro udržení soukromí uživatelů komunikačních systémů. Cílem práce je rozbor současných metod, jejich srovnání a zhodnocení jejich použitelnosti pro praktickou implementaci. V praktické části se bude implementovat vybraný systém ve vhodném prostředí.

Na začátku práce budou uvedeny jednotlivé systémy a prvky zejména z oboru kryptografie s veřejným klíčem (tzv. asymetrické kryptosystémy) mezi které patří například algoritmus RSA, protokol o výměně klíčů DH, digitální podpisy, šifrovací systém ElGamal, atd. Dále budou uvedeny protokoly a metody identifikace či autentizace zaměřené především na problém anonymní autentizace a koncept nulové znalosti (v ang. zero-knowledge concept). V druhé části budou analyzovány skupinové podpisy (v ang. Group signature) a systémy elektronických hotovostí (v ang. e-cash) používající elektronické mince (v ang. e-coin). Třetí část představí anonymní autentizační systémy AAS, ze kterých vychází praktická část.

Poslední částí této práce je popis praktické implementace v prostředí .NET v jazyce C#, kde budou popsány funkce jednotlivých implementovaných aplikací a rozbor jejich bezpečnosti a celkové efektivity.

V závěru pak bude shrnuto použití a srovnání analyzovaných systémů a protokolů a jejich zhodnocení spolu s výsledky z praktické části.

1 KRYPTOGRAFIE

Kryptografie je studium technik a prostředků, které se zabývají utajováním zpráv. Cílem je utajování smyslu obsahu zpráv převodem do podoby, která je čitelná jen se speciální znalostí. Mezi služby zjišťované kryptografickými prostředky patří důvěrnost, integrita, autentičnost zpráv a nepopiratelnost. Tyto techniky bývají založeny na obtížnosti řešení matematických problémů nebo na časové náročnosti hledání řešení.

V první kapitole budou popsány jednotlivé systémy a prvky z kryptografie, ze kterých se pak skládají systémy pro autentizaci a pro ochranu soukromí na Internetu. Jinými slovy zde budou představeny pouze systémy, které nějakým způsobem souvisí s anonymní autentizací tj. skupinovými podpisy, elektronickými mincemi atd. Proto budou vynechány například symetrické systémy či metody autentizace na základě identity uživatele a jiné systémy, které samozřejmě do kryptografie patří, ale jsou v této práci bezpředmětné. Více o těchto systémech pak v literaturách [20], [37], [58] a [68].

1.1 Základní terminologie a značení prvků

Zde jsou uvedeny některé definice, termíny a označení používané v kryptografii a matematice. Termínů je obecně v kryptografii, matematice či bezpečnosti IS celá řada, více pak v [20], [58] a [68]. Označení prvků je nutné při popisu nejrůznějších algoritmů a matematických vztahů. Označení se v české a anglické literatuře v některých případech mírně liší.

1.1.1 Pojmy a prvky z hlediska kryptografie

Abeceda zprávy A

Množina symbolů, které se mohou při kódování (převodu zpráv a klíčů na vhodný tvar) nebo při šifrování použít. Nejčastěji používaná je binární abeceda, kdy množina $A = \{0,1\}$.

Množina zpráv Z (v ang. M)

Množina všech možných zpráv. Jedná se o řetězec symbolů definovaných abecedou zprávy. Množina zpráv obsahuje několik elementů, které nazýváme otevřený text (v ang. plaintext). Segment zprávy značíme *z* či *m*.

Šifrování E

Proces transformace otevřeného textu na zašifrovaný text za účelem skrytí smyslu obsahu zprávy.

Zašifrovaný text či kryptogram C

Jedná se o řetězec symbolů, které jsou definované abecedou znaků (která může být odlišná od abecedy znaků definující otevřený text) a který vzniká po použití procesu šifrování na otevřený text $C = E(Z, K_E)$. Jeden segment kryptogramu značíme *c*.

Dešifrování D

Proces transformace zašifrovaného textu na otevřený text. Jedná se o inverzní proces k šifrování. Obnovuje původní zprávu $Z = D(C, K_D)$.

Šifrovací klíč K_E (v ang. K_e)

Jedná se zpravidla o číselné parametry, či řetězec bitů o délce *K*. Používá se při procesu transformace otevřeného textu na zašifrovaný text, čili šifrování. Popřípadě značíme *e*.

Dešifrovací klíč K_D (v ang. K_d)

Jedná se zpravidla o číselné parametry či řetězec bitů o délce *K*. Používá se při procesu transformace zašifrovaného textu na otevřený text, čili dešifrování. Popřípadě značíme *d*.

1.1.2 Pojmy a prvky z hlediska matematiky

Abelovská grupa

Grupa je abelovská, pokud platí komutativnost pro všechny prvky z grupy, tj. $a+b=b+a$.

Bilineární mapování

V matematice je bilineární mapování určitá funkce kombinace prvků dvou vektorových prostorů dávající prvky třetího vektorového prostoru tak, že jsou argumenty vůči sobě lineární. Příkladem je násobení matic $\mathrm{M}(m,n) \times \mathrm{M}(n,p) \rightarrow \mathrm{M}(m,p)$.

Cyklická grupa

Grupa, která může být generována jedním generátorem α či více generátory. To znamená, že v grupě existuje prvek α tak, že každý prvek grupy je mocninou α.

Grupa

Je algebraická struktura, tj. množina spolu s binární operací splňující tyto matematické axiomy: uzavřenost, asociativitu, existenci neutrálního prvku a existenci inverzních prvků. Podle binární operace nad grupou nazýváme neutrální prvek nulovým nebo jednotkovým.

Izomorfismus

Matematický pojem, který představuje určité zobrazení mezi dvěma množinami, pomocí kterého můžeme mezi nimi beze ztráty jakékoliv informace libovolně přecházet. Jinými slovy, každému prvku jedné množiny odpovídá právě jeden prvek druhé množiny a to včetně zachování veškerých jeho strukturálních vlastností (vztahů k ostatním prvkům), více v [58].

Konečné pole

V abstraktní algebře konečné pole je pole obsahující pouze konečný počet prvků.

Okruh

Je algebraická struktura s distributivností, která je vzhledem ke sčítání komutativní grupou a vzhledem k násobení pologrupou.

Pole

V abstraktní algebře je pole těleso, které je vzhledem k násobení komutativní grupou.

Pologrupa

V algebře je pologrupa algebraická struktura s jednou asociativní binární operací.

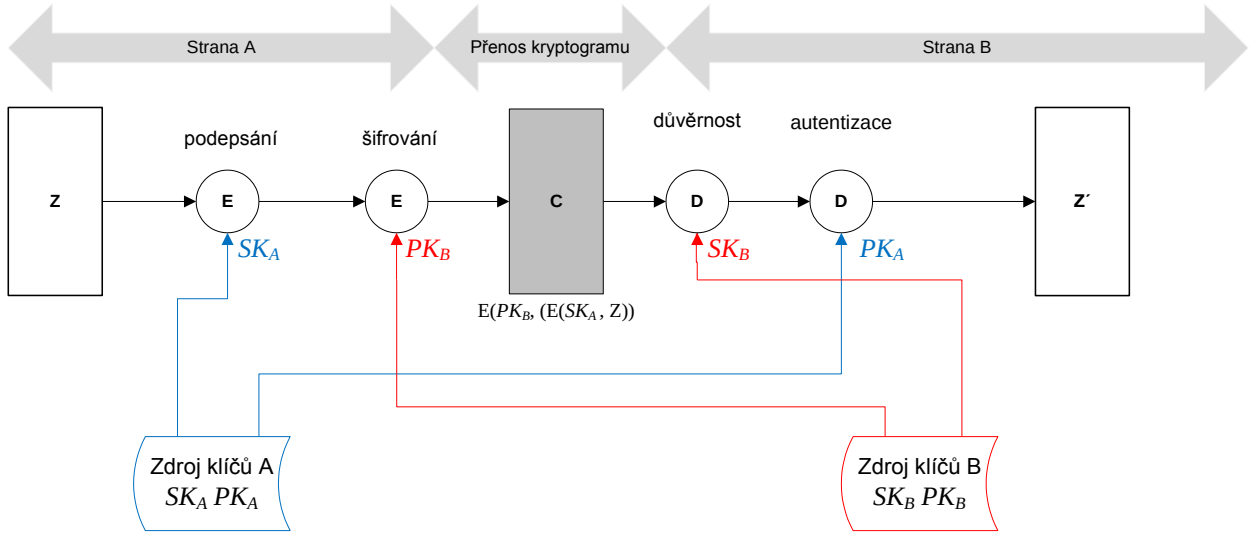
Těleso

Je algebraická struktura, na které jsou definované dvě binární operace. Oproti okruhu přináší existenci inverzního prvku pro obě binární operace.

1.2 Kryptografické systémy s veřejným klíčem

Kryptografické systémy s veřejným klíčem (asymetrické kryptosystémy) se od kryptografických systémů s tajným klíčem (symetrické kryptosystémy) liší v mnoha ohledech. Základní rozdíl spočívá v držení klíčů. U symetrického kryptosystému se musí šifrovací i dešifrovací klíč držet v tajnosti, neboť jsou tyto klíče stejné $K_E = K_D$ či snadno od sebe odvoditelné, kdy je funkce f výpočetně realizovatelná v rozumném čase $K_D = f(K_E)$. Naopak u asymetrického systému používáme různý šifrovací a různý dešifrovací klíč. Funkce $K_D = f(K_E)$ či $K_E = f^{-1}(K_D)$ jsou výpočetně neproveditelné. Za předpokladu této skutečnosti, lze pak tajit pouze jeden klíč tzv. soukromý *SK* (v ang. private key) a druhý pak používat veřejně tzv. veřejný klíč *PK* (v ang. public key).

Zdali utajovat šifrovací, nebo naopak dešifrovací klíč, pak záleží pouze na tom, jedná-li se o systém zajišťující důvěrnost, nebo autentičnost. Jestliže pak veřejným klíčem *PK* zašifruje zprávu kdokoliv, může pak ověřit důvěrnost a integritu zprávy pouze majitel soukromého dešifrovacího klíče *SK*. Pouze majitel soukromého klíče dokáže přečíst zprávu, která byla zašifrovaná veřejným klíčem. Pokud bychom zaměnili smysl procesu, šifrovacím klíčem K_E se stane soukromý klíč *SK*. Naopak dešifrovacím klíčem K_D se stane veřejný klíč *PK*. Po zašifrování zprávy *Z* soukromým klíčem *SK* zajistíme její autentičnost. Tvrzení, že tento kryptogram *C* může vytvořit pouze majitel soukromého klíče, může ověřit kdokoliv s veřejným klíčem $PK = K_D$. Tato skutečnost se využívá u digitálních podpisů. Obr. 1 zobrazuje schéma systému, který zajišťuje důvěrnost i autentičnost zprávy.



Obr. 1: Kryptosystém s veřejnými klíči zajišťující důvěrnost a autentičnost.

Výhodou asymetrických kryptosystémů je jednodušší distribuce klíčů. Nevýhodou je pomalé šifrování a dešifrování. Proto se tyto systémy výhodně používají při výměně klíčů, přenosu klíčů pro symetrické systémy, šifrování krátkých zpráv, podepisování zpráv a zajištění autentizace entit (dat nebo osob).

Funkčnost asymetrického kryptosystému se opírá o tzv. jednocestnou funkci (v ang. trapdoor one-way function) definovanou jako $y = f(x)$, kde výpočet *y* je relativně snadná úloha, avšak výpočet argumentu $x = f^{-1}(y)$ (tj. ze znalosti hodnoty funkce f) je výpočetně prakticky nemožný. V současné době se z matematického oboru teorie čísel využívá především problému faktorizace čísla a problému diskrétního logaritmu, ale také existují techniky typu problém kvadratických zbytků (v ang. quadratic residuosity problem), počítání kořenů druhé mocniny

(v ang. computing square roots), problém batohu (v ang. knapsack - subset sum problem) či faktorizace polynomů nad konečnými poli. Podrobný popis lze vyhledat v [58]. Kromě výše zmíněných problémů se v asymetrických kryptosystémech uplatňují i kryptosystémy založené na eliptických křivkách, o kterých je více v [58] a [68].

1.2.1 Problém faktorizace celého čísla a RSA problém

Bezpečnost mnoha kryptografických technik závisí na problému faktorizace velkých celých čísel (v ang. IF, Integer Factorization). Mezi techniky, které jsou na tomto problému založeny, uveďme například schéma šifrovaní veřejným klíčem RSA, podpisové schéma RSA či schéma šifrování veřejným klíčem Rabin. Někdy se tyto systémy nazývají jako faktorizační systémy. Následující sekce shrnuje definice a aktuální poznatky o algoritmech pro problém faktorizace celého čísla. Definice jsou pro lepší přehlednost zaměřeny na práci s dekadickými čísly. V praxi se pak číselné hodnoty či jiné symboly převádí na binární řetězce v binární abecedě, se kterými pak dále kryptosystém pracuje.

Definice problému faktorizace celého čísla (zkr. Faktorizace)

Každé kladné celé číslo n, lze rozložit na součin prvočísel p_i (tj. faktorizovat) určitých mocnin $n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$, kde p_i jsou různá prvočísla o mocnině $e_i \geq 1$, viz [58].

Obecně je známo, že úloha testování, zdali se jedná o obyčejné složené kladné celé číslo, nebo naopak o prvočíslo, je mnohem lehčí než problém faktorizace. Tato úloha je řešitelná v polynomiálním čase, proto se testuje před samotnou faktorizací, zdali se skutečně jedná o složené kladné celé číslo. Tato úloha se nazývá jako test prvočíselnosti.

V literatuře [58] je uveden postup tzv. rozdělování (v ang. splitting), který se snaží rozdělit n na celá čísla a a b, kde $n = ab$, které nemusí být nutně prvočísla. Poté se a i b otestují testem prvočíselnosti, v případě potvrzení neprvočíselnosti je znovu rekurentně rozkládat na další složky. Takto lze nakonec získat faktorizaci n.

Speciální algoritmy pro řešení faktorizace *n*

Tyto speciální algoritmy jsou účinné a efektivní pouze za předpokladu určitých vlastností čísla n. Naopak pokud zvolíme nevhodný algoritmus, pak může být jeho činnost velmi neefektivní. Jedná se například o proces dělení, Pollard rho algoritmus, Pollard p-1 algoritmus, algoritmus eliptických křivek, atd. V kontrastu se speciálními algoritmy jsou takzvané základní algoritmy pro účel faktorizace, závislé pouze na velikosti n, které mají určitou provozní dobu. Mezi základní faktorizační algoritmy patří Kvadratické síto (QS - Quadratic Sieve) či Síto číselného pole (NFS - Number Field Sieve). Obecně se doporučuje používat speciální algoritmy za specifických podmínek. Pokud cyklus rozkladu selže, nasadíme základní algoritmus. Bližší informace o činnostech speciálních a základních algoritmů pro řešení faktorizace lze najít v [58].

V roce 2005 se podařilo rozložit číslo n o velikosti 200 decimálních číslic, tj. 663 dlouhý bitový řetězec, pomocí distribučních metod, tzv. algoritmu Mřížkového síta (v ang. Lattice sieve), bližší přehled v [68].

Definice RSA problému

Pro vyřešení RSA problému máme k dispozici kladné celé číslo n, které je produktem dvou odlišných lichých prvočísel p a q ($n = pq$), dále známe kladné celé číslo e, které je nesoudělné s n a tvoří největší společný dělitel $\gcd(e,(p-1)(q-1)) = 1$ a celé číslo c. Úkolem je najít celé číslo m

pro které platí $m^e \equiv c \pmod{n}$, dle definice viz [58]. Podmínky u vzniku parametrů n a e zajišťují, že pro každé $c \in \{0,1,\ldots,n\text{-}1\}$ vzejde právě jedno $m \in \{0,1,\ldots,n\text{-}1\}$, proto $m^e \equiv c \pmod{n}$ a jedná se o permutační hledání řešení funkce $f(m) = m^e \bmod n$, viz [58].

Můžeme identifikovat tři matematické přístupy k útočení na RSA, viz [68]:

- Faktorizace n na dvě prvočísla p a q dovoluje výpočet Eulerovy funkce $\varphi(n) = (p - 1)(q - 1)$. Díky této hodnotě pak lze determinovat dešifrovací klíč $d \equiv e^{1} \pmod{\varphi(n)}$.
- Determinace přímo Eulerovy funkce $\varphi(n)$ bez determinování p a q. Postup pak stejný jako u prvního přístupu.
- Determinace d přímo, bez determinování Eulerovy funkce $\varphi(n)$.

1.2.2 Kryptosystém RSA

Kryptosystém RSA je první asymetrickým kryptosystémem s veřejným klíčem, který je vhodný jak pro podepisování, tak i pro šifrování. Název RSA značí iniciály autorů Rivest, Shamir a Adleman. Kryptosystém RSA tvoří základní stavební prvek pro mnoho systémů s ochranou soukromí a také v implementovaném systému AAS v praktické části.

Použití

RSA se používá zejména k podepisování zpráv, tj. zaručení autentičnosti zpráv (osob a jiných entit) a k šifrování krátkých zpráv, které přenáší klíče pro rychlejší symetrické systémy, či pro přenos hesel při autentizaci uživatelů.

Princip

Odolnost vůči kryptoanalýze je založena na problému faktorizace velkého celého čísla na prvočísla, který je popsán v sekci 1.2.1. Bezpečnost RSA je zaručena skutečností, že faktorizace velkých čísel používaných v RSA $n = 2^{1024}$ až 2^{2048} je v dnešní době prakticky nereálná. Doposud neexistuje efektivní algoritmus pro faktorizaci velkých čísel, který by pracoval v polynomiálním čase. Naopak násobení velkých čísel je relativně snadná úloha.

Konstrukce

Každá entita v kryptosystému RSA vytvoří veřejný klíč a k tomu korespondující soukromý klíč dle následujících kroků:

- generování dvou náhodných velkých a odlišných prvočísel p a q přibližně stejné délky, v praxi obvykle 2^{256},
- výpočet čísel: $n = pq$ a Eulerovy funkce (pro prvočísla se její výpočet podstatně zjednoduší) $\varphi(n) = (p - 1)(q - 1)$, kde číslo n se nazývá modulus,
- zvolení náhodného celého čísla e, představující veřejný šifrovací klíč (šifrovací exponent), z rozsahu $1 < e < \varphi(n)$, a které je nesoudělné s $\varphi(n)$,
- pomocí rozšířeného Euklidova algoritmu nalezneme jedinečné celé číslo d, které představuje soukromý dešifrovací klíč (dešifrovací exponent), podle vztahu $ed \equiv 1 \pmod{\varphi(n)}$, tj. $d = e^{-1} \bmod \varphi(n)$,
- parametry e a n zveřejníme a ostatní parametry zůstanou utajeny.

V praxi se pak při konstrukci či generování klíčů RSA vyskytuje hned několik podproblémů např. generování náhodných prvočísel a jejich testování, více v [68], dále správná volba šifrovacího klíče *e*, která má vliv na rychlost šifrování a bezpečnost kryptosystému atd.

V neposlední řadě je problém distribuce veřejných klíčů, kdy je potřeba vůči útoku třetí osobou tj. mužem uprostřed (v ang. man-in-the-middle), zajistit vhodný obranný mechanismus, například využít certifikáty v infrastruktuře veřejných klíčů PKI (od ang. public key infrastructure). Podrobněji o dané problematice v [37], [58] a [68].

Postup šifrování a dešifrování pro mód šifrování zpráv

Níže je uvedeno schéma, kdy strana *B* šifruje zprávu *Z* pro stranu *A*:

- Nejprve *B* získá veřejný klíč, který je složen ze šifrovacího exponentu *e* a modulu *n*, od strany *A*, která tento klíč vygenerovala a drží v utajení svůj dešifrovací klíč a ostatní parametry.
- Strana *B* zprávu *Z* rozdělí na bloky z_i o stejné délce, za podmínky $z_i < n$. Každý *i*-tý blok se chápe jako číslo z_i. Při uvažování binární abecedy to znamená, že při délce $n = 2^{1024}$, budou jednotlivé bloky zprávy kratší než 1024 bitů.
- Následně šifruje zprávu *i*-krát blok po bloku dle $c_i = z_i^e \bmod n$.
- Z jednotlivých segmentů kryptogramů c_i poskládá výsledný kryptogram *C* a posílá adresátovi *A*.

Strana *A* po obdržení kryptogramu *C* provede dešifrování:

- Nejprve se kryptogram *C* rozdělí na jednotlivé bloky c_i.
- Každý blok dešifruje pomocí soukromého klíče *d* $z_i = c_i^d \bmod n$.
- Výsledná zpráva *Z* je poskládána z bloků z_i.

Každý může vytvořit zašifrovanou zprávu pomocí veřejného klíče *e* a modula *n* strany *A*, ovšem přečíst (dešifrovat) ji dokáže pouze majitel soukromého klíče *d* čili strana *A*. Důkaz o funkčnosti dešifrovacího schématu lze najít v [68].

Postup šifrování a dešifrování pro mód podepisování zpráv

Zde je uvedeno schéma, kdy strana *A* podepisuje zprávu *Z* pro stranu *B*:

- Libovolně v pořadí před dešifrováním získá *B* veřejný klíč, který je složen ze šifrovacího exponentu *e* a modulu *n*, od strany *A*, která tento klíč vygenerovala a drží v utajení svůj dešifrovací klíč a ostatní parametry.
- Strana *A* zprávu *Z* rozdělí na bloky z_i o stejné délce, za podmínky $z_i < n$. Každý *i*-tý blok se chápe jako číslo z_i.
- Dále šifruje zprávu *i*-krát blok po bloku svým soukromým klíčem dle $c_i = z_i^d \bmod n$.
- Z jednotlivých segmentů kryptogramů c_i poskládá výsledný kryptogram *C* (podepsaná zpráva) a posílá adresátovi *B*.

Strana *B* po obdržení kryptogramu *C* provede dešifrování:

- Nejprve kryptogram *C* rozdělí na jednotlivé bloky c_i.
- Každý blok dešifruje pomocí veřejného klíče *e* strany *A*, $z_i = c_i^e \bmod n$.
- Výsledná zpráva *Z* je poskládána z bloků z_i.

Strana *B* a rovněž kdokoliv jiný, kdo zná veřejný klíč *e* a modulus *n* strany *A*, si může ověřit, zdali je zpráva autentická, tedy původní, jelikož mohla být zašifrována pouze soukromým klíčem *A*, korespondujícím pouze s veřejným klíčem *A*.

V opačném směru platí tzv. nepopiratelnost strany *A*, kde strana *A* nemůže popřít podepsání zprávy.

1.2.3 Problém diskrétního logaritmu a Diffie-Hellman problém

Problém diskrétního logaritmu se řadí spolu s problémem faktorizace velkých celých čísel mezi výpočetně prakticky neřešitelné, a proto jsou vhodně využity v kryptografii. Logaritmické veřejné systémy (v ang. DL, Discrete Logarithm), mezi které patří protokol Diffie-Hellman a jeho deriváty, šifrovací systém ElGamal a ElGamal podpisové schéma, jsou založeny na problému nalezení diskrétního logaritmu. Tyto šifrovací a podpisové systémy tvoří základní části systémů s ochranou soukromí. Na problému diskrétního logaritmu stojí fáze autentizace implementovaného systému AAS v praktické části.

Obecně problém diskrétního logaritmu (v a.j. DLP – Discrete Logarithm Problem)

Jestliže máme funkci $y = f(x) = g^x \bmod p$, kde velké prvočíslo p a základ mocniny g jsou známy, potom lze snadno vypočítat hodnoty y pro argument x. Avšak inverzní výpočet, nalezení hodnoty x pro dané y, je výpočetně prakticky nemožný.

Definice problému diskrétního logaritmu pro cyklické grupy

Při základní formulaci problému diskrétního logaritmu pro konečné cyklické grupy, více o grupách [58], pak můžeme uvažovat, že pro danou konečnou cyklickou grupu G o řádu n, generátoru α grupy G a prvku $\beta \in G$ je nalezení parametru x, $0 \leq x \leq n\text{-}1$ dle $\alpha^x = \beta$ těžko řešitelný problém.

Řešení problému diskrétního logaritmu v cyklické grupě G řádu n je v podstatě počítání izomorfismu mezi G a $\mathbb{Z}_n$, více v [58].

Algoritmy pro řešení problému diskrétního logaritmu

Algoritmy, obdobně jako u problému faktorizace celého čísla, jsou rozděleny na základní, které jsou široce použitelné, a na speciální, které jsou efektivnější za jistých podmínek. Mezi základní algoritmy, které pracují s libovolnými grupami, patří algoritmus vyčerpávajícího hledání, který spočívá v prostém počítání $\alpha^0, \alpha^1, \alpha^2, \ldots$ dokud nezískáme hodnotu β. Z kryptografického hlediska je tento algoritmus nepoužitelný vzhledem k velkým řádům n. K dalším algoritmům se řadí např. algoritmus malého a velkého kroku (v ang. baby-step giant-step) a Pollardův rho algoritmus.

Mezi speciální algoritmy patří např. Pohling-Hellmanův algoritmus či index-kalkul algoritmus, který je efektivní pouze v určitých grupách o jistých parametrech. Algoritmy jsou přehledně popsány v [58].

Problém Diffie-Hellman

Známe prvočíslo p a generátor (základ mocniny) α ze $\mathbb{Z}_p^*$ a prvky $\alpha^a \bmod p$, $\alpha^b \bmod p$ a hledáme hodnotu $\alpha^{ab} \bmod p$. Zobecněno na cyklické grupy, pak známe konečnou cyklickou grupu G, generátor α z G a elementy grupy α^a, α^b, kde hledáme α^{ab}.

Složitost DH problému je naznačena v následující úvaze. Předpokládejme, že existuje nějaký efektivní algoritmus řešení diskrétního logaritmu, díky kterému bychom byli schopní vypočítat parametr a z rovnice $\alpha^a \bmod p$. Ostatní parametry již známe, čili stačí vypočítat rovnici $(\alpha^b)^a = \alpha^{ab} \bmod p$, viz [58].

1.2.4 Kryptosystém ElGamal

ElGamal se používá v asymetrické kryptografii podobně jako RSA, ale v menší míře, jelikož jeho funkce šifrování expanduje velikost zprávy Z na kryptogram C v poměru 1:2, čili kryptogram je o dvojnásobné velikosti vůči zprávě. Naopak jeho výhodou je, že je volně dostupný jako svobodný software a je používán v rámci bezpečnostního souboru protokolů a schémat OpenPGP (Pretty Good Privacy).

Podobně jako RSA i ElGamal se dělí na dvě funkční větvě: podpisové schéma ElGamal a šifrování ElGamal. Často se používá v rámci hybridního systému, kdy slouží pouze k šifrovanému přenosu klíčů pro symetrický kryptosystém.

Bezpečnostní princip je založen na složitosti diskrétního logaritmického problému a Diffie-Hellman problému. Princip, konstrukce a další podrobnosti jsou popsány v [58].

1.3 Obecné kryptografické prvky

Tato část krátce popisuje vybrané obecné kryptografické prvky jako například autentizační funkci, hašovací funkce, autentizaci entit, digitální podpis, certifikáty a infrastrukturu veřejných klíčů PKI. Tyto prvky tvoří základní stavební bloky v systémech s ochranou soukromí.

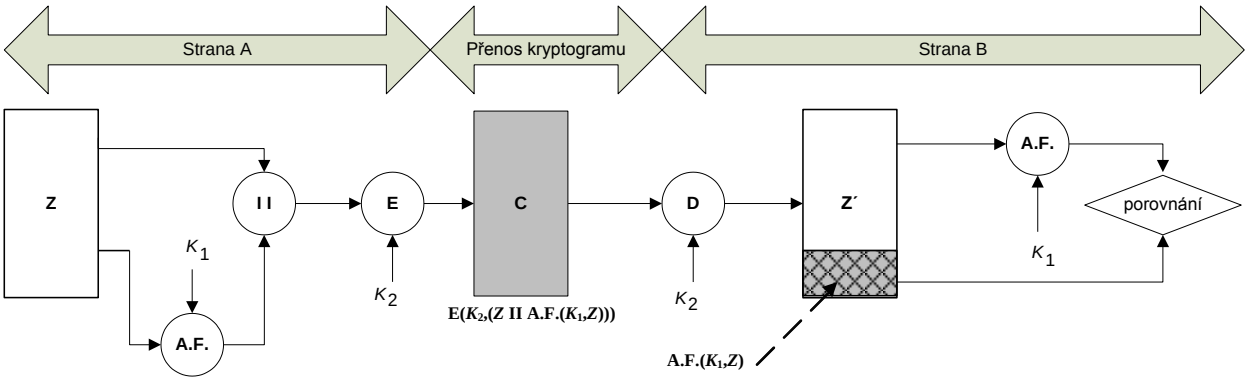
1.3.1 Autentičnost dat

Autentičnost dat můžeme zajistit například digitálním podpisem, např. pomocí podpisového schématu RSA. Zjednodušeně jde o zašifrování zprávy soukromým klíčem strany *A*, která podepíše zprávu a zaručuje její původnost. Ověřit podpis dat pomocí veřejného klíče strany *A* může kdokoliv. Takový způsob má však dva zásadní nedostatky. Prvním nedostatkem je pomalá funkce šifrování a dešifrovaní asymetrického kryptosystému vzhledem k objemnosti zpráv. Řešením je používat hašovací funkce (vytvoření otisku zprávy). Druhý nedostatek přináší problém důvěry, zdali veřejný klíč patří skutečně požadované straně a ne útočníkovi, který se za danou stranu vydává. Proto se buduje infrastruktura veřejných klíčů PKI, která vydává certifikáty a které účastníci v systému důvěřují, tzv. třetí strana. Obecně schémata PKI a její deriváty, využívají certifikáty, které obsahují informace o uživateli, tudíž jsou neanonymní. Proto v této práci nebudou blíže rozebrány a více informací o PKI v [37].

Autentizační funkce

Jedná se o funkci, která vytváří autentizační kód h ze zprávy z v závislosti na nějakém tajném klíči (v ang. MAC Message Authentication Code). Výhodou této funkce je, že zprávu může vytvořit pouze majitel tajného klíče. Algoritmus MAC nemusí být vratný, čili jedná se o funkci, která transformuje zprávu o libovolné délce na malou pevnou délku dat (many-to-one). Z hlediska symetrické kryptografie zprávu pak může ověřit pouze opět uživatel, který zná tajný klíč a vypočítá znovu autentizační kód h' přijaté zprávy z'. Pokud platí, že původní kód h připojený ke zprávě, nebo který byl uložen v kryptogramu, se rovná vypočtenému h' pak lze prohlásit, že zpráva je autentická, čili nepozměněná. Takto se zajišťuje integrita neboli

skutečnost, že zpráva nebyla pozměněna během přenosu. Navíc se bezpečnost zprávy může posílit šifrováním, které může být umístěno po vytvoření aut. kódu, více informací v [68]. Principální schéma je znázorněno na Obr. 2.



Obr. 2: Autentizace zprávy *z* a její šifrování. Zpráva *z* a autentizační kód *h* se vytvoří před zašifrováním.

1.3.2 Autentizace entit

Autentizace je proces ověřující identitu entity. Entity tvoří např. procesy, osoby, služby či různá zařízení. Proces autentizace může být jednostranný či vzájemný. Obecně se autentizace dělí na tři základní skupiny. Autentizace znalostí, vlastnictvím (předmětem) či biometrikou. Tyto skupiny je možné vhodně kombinovat a zesílit tak proces autentizace. Jelikož je Internet velký datový konglomerát, bývá nejelegantnější a nejpraktičtější používat autentizaci na základě nějaké znalosti. Autentizace entit z pohledu znalosti informace používá tři mechanismy.

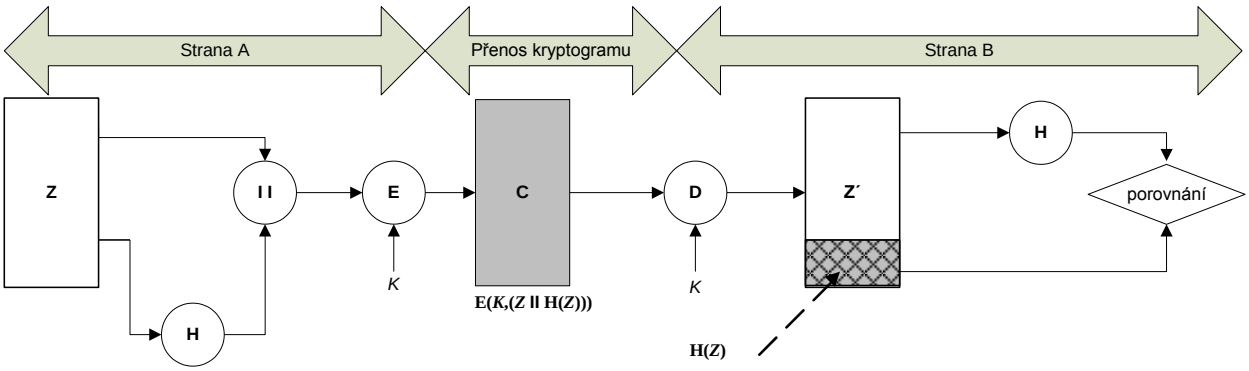
Prvním a nejslabším mechanismem je použití hesla, tedy přímé prokázání znalosti. To však obnáší jeden základní problém, a sice ten, že žadatel *A* posílá klíčovou znalost ověřovateli *B*, pokud je ověřovatel *B* nečestný či falešný, pak může bez problému použít heslo žadatele *A* a tak sám využívat jeho služby. Dalším problémem je přihlašování na veřejných místech, kde heslo na obrazovce či jeho zadávání na klávesnici může být útočníkem, či optickým zařízením (např. průmyslovou kamerou) zpozorováno.

Druhým již silnějším způsobem je využití procesu výzva-odpověď. Žadatel *A* neprozrazuje přímo svoji znalost, ale pouze potvrzuje vhodnou odpovědí ověřovateli *B* znalost reakce na výzvu. Zde se pak setkáváme s útoky typu záměrné předkládání výzev od nečestného ověřovatele *B*, kdy na základě odpovědí může určit chráněnou znalost *A*. Stejně jako autentizace heslem je tento způsob náchylný také na útok muže uprostřed nebo na útok se zvoleným textem.

Z výše uvedených nedostatků dnešních běžných autentizačních metod se začíná uvažovat o třetím způsobu, který spočívá v podání důkazu o jisté znalosti a při tom se nevyzrazuje žádný kus chráněného tajemství či informace. Jinými slovy je žadatel schopen prokázat znalost svého tajemství, bez toho aby odhaloval nějaké informace svého tajemství. Tímto může ověřovaná entita zůstat vůči ověřovateli v anonymitě. Tato metoda autentizace je založena na konceptu nulové znalosti a umí zachovat anonymitu uživatele (tzv. anonymní autentizace). Podrobnější informace o konceptu nulové znalosti (v ang. zero-knowledge) v [35]. Anonymní autentizace má za úkol chránit soukromí uživatele a přitom mu poskytnout bezpečný přístup ke službám, datům, místům a jiným aktivům, které smí uživatel oprávněně používat.

1.3.3 Hašovací funkce

Hašovací funkce (v ang. hash) je jednocestná kompresní funkce, která stejně jako aut. kód zprávy, produkuje ze zprávy proměnné délky, kód o pevné délce (256, 384 či 512 bitů), tzv. otisk, který zaručuje integritu dat. Hašovací kód poskytuje detekci změny jakéhokoliv bitu ve zprávě *z*. Na rozdíl od funkce MAC, hašovací funkce nepoužívá tajný klíč. Skutečnosti, že vypočtené haše může porovnat kdokoliv, lze využít u veřejných kryptosystémů, respektive u digitálních podpisů. Základní princip spolu se šifrováním viz schéma na Obr. 3.



Obr. 3: Výpočet haše ze zprávy *z* a jeho přenos v kryptogramu *c*.

Požadavkem je, aby funkce byla jednocestná, čímž zaručuje praktickou nemožnost konstrukce jiné zprávy *z´* z daného haše *h* vypočteného pro zprávu *z*. Druhým požadavkem je tzv. bezkoliznost, tj. pravděpodobnost nalezení dvou zpráv produkující stejný hašovací kód je prakticky nulová. Hašovací algoritmus v podstatě provádí iterativně nejrůznější sady blokových operací, zahrnující také různé typy operací modulární aritmetiky a logické binární operace. Více o funkčnosti jednotlivých operací v [68]. Nejběžnější algoritmus je MD5, ale nejbezpečnějšími algoritmy jsou SHA-512 a algoritmus Whirlpool. Algoritmus SHA-512 je nasazen taky v implementovaných aplikacích v praktické části.

Algoritmus SHA-512

Hašovací algoritmus SHA-512 vytváří 512 bitů dlouhý výstupní otisk zprávy. Před samotným algoritmem se provedou následující kroky:

- Rozdělení a výplň zprávy bity tzv. vyplnění (zpráva je shodná s 896mod1024).
- Přidání 128-bitového řetězce udávající délku zprávy.
- Inicializace bufferů. Buffer má velikost 512 bitů a je dělen na osm 64-bitových registrů.
- Samotný algoritmus obsahuje 80 kol, které se provedou v jedné iteraci. Každé kolo má na vstupu 512-bitový buffer, který pak činí výstup předchozího kola, dále zpracovává 64 bitový úsek (tedy jedno slovo bloku) determinovaný z 1024-bitového bloku zprávy a 64-bitovou přídavnou konstantu, která má eliminovat pravidelnosti vstupních dat. Výstup kola se připojuje na vstup následujícího kola, který přidává opět další hodnoty ze zprávy (další slovo-64 bitů) a konstantu. Výstup posledního 80 kola se sčítá s hodnotami z předchozí iterace. Samotný algoritmus je podrobně popsán v [68].

Další hašovací algoritmus Whirlpool, který je založen na použití blokových šifer, je ve své funkčnosti podobný kryptosystému AES. Na vstupu se přidávají bloky zprávy o velikosti 512 bitů, které jsou podobně předem upravovány jako u SHA (vyplnění, údaj o délce zprávy). Na

rozdíl od bufferů, se zde používají hašovací matice o velikosti 8x8 bajtů. Samotný proces je pak proveden v 10 kolech. Při každém kole se provádějí substituční, posuvné, mixující a logické operace. Výstup procesu se přenáší jako vstup do další iterace. Po provedení posledního procesu u posledního bloku se na výstupu objeví 512 bitů dlouhý hašovací kód (otisk). Více informací v [68].

1.3.4 Digitální podpis

Digitální podpis přebírá vlastnosti klasického podpisu. Obvykle digitální podpis funguje na principu vypočtení otisku h ze zprávy Z, potom se tento haš zašifruje a pošle se jako podpis $P = \mathrm{E}(h,x)$ spolu se zprávou Z příjemci. Příjemce po obdržení zprávy Z' vypočte znovu haš h' a veřejným klíčem podepisujícího dešifruje jeho podpis, který ukrývá haš h. Pokud se otisky $h' = h$ shodují (tj. $h' = \mathrm{H}(Z) = \mathrm{D}(P',y)$), tak zpráva a podpis P přišli nepozměněny. Digitální podpis pomocí kryptosystému RSA je nasazen v implementovaném systému v praktické části.

Samotné digitální podpisy nedokážou ověřit, zdali podepisující osoba a její vydávaný veřejný klíč je skutečně požadovaná osoba/entita s příslušným veřejným klíčem a tedy ne nějaký útočník, který se za osobu/entitu vydává. Tuto skutečnost obvykle řeší certifikáty v rámci tzv. infrastruktury veřejných klíčů.

1.3.5 Certifikáty a PKI

Abychom zabránili podvodným podpisům a předstírání identity, zavedly se tzv. certifikáty, které jsou v podstatě elektronicky podepsané tvrzení, které dosvědčují vlastnictví, že jistá osoba vlastní jistý veřejný klíč. Tyto certifikáty vydává nějaká důvěryhodná strana, které účastníci důvěřují a také musí respektovat její autoritu při řešení sporů. Tato instituce se nazývá certifikační autorita CA a její veřejný klíč V_{CA} je bezpečně doručen všem uživatelům, kteří pak pomocí tohoto klíče ověřují jakékoli certifikáty osob vydané stejnou CA.

Při registraci do takového systému infrastruktury veřejných klíčů PKI se uživatel nejprve ověří, fyzicky se prokazuje jeho totožnost a pravdivost údajů o dané osobě u jisté registrační autoritě RA, která zpracuje jeho žádost. V žádosti F_A musí být uveden již jeho veřejný klíč V_A, jeho identifikační údaje ID_A a další režijní data o certifikátu DD_A (doba platnosti atd.). RA po ověření korektnosti žádosti, pak předá žádost F_A již ve formě speciálního formuláře CA a ta formulář svým tajným klíčem T_{CA} podepíše, čímž vznikne podpis $P_A = \mathrm{E}(\mathrm{H}(F_A),T_{CA})$. Certifikát $CRT_A = F_A \,\|\, P_A$ předá zpět registrační autoritě a ta jej předá uživateli. Tímto pak nabízí ostatním uživatelům bezpečný prostředek k autentizaci jeho vlastního veřejného klíče. Lze si povšimnout, že certifikát obsahuje v části F_A i informace o uživateli ID_A, čili tento způsob autentizace je neanonymní, více informací viz [37]. V implementovaném systému AAS využíváme certifikaci standardu X.509.

1.4 Kryptografické prvky v anonymní autentizaci

V této části budou popsány kryptografické prvky, které se používají v anonymní autentizaci a při stavbě systémů s ochranou soukromí uživatelů.

1.4.1 Autentizace na konceptu nulové znalosti

Autentizace s protokoly s nulovou znalostí (ZKIP) fungují na bázi série úkolů a odpovědí. Z toho plyne, že možnost neoprávněného přístupu je pak dána pouze jistou pravděpodobností, která

závisí na velikosti množiny úkol-reakce. Žadatel tak má za úkol přesvědčit ověřovatele o znalosti tajemství prostřednictvím správných reakcí na úkoly. V podstatě se jedná o násobné zprávy výzva-odpověď. Není tedy absolutní garance bezpečnosti.

Obecná struktura protokolu s nulovou znalostí

Protokol s nulovou znalostí lze přirovnat k mechanismu rozkrojení koláče. Jeden účastník rozděluje koláč a druhý na oplátku zvolí díl. Tímto dojde ke spravedlivému rozdělení. Zde si tedy žadatel *A* určí náhodný prvek z tajného ujednání a z toho vypočte příslušné veřejné osvědčení o znalosti. V podstatě si definuje soubor otázek, pro které zná odpověď. Protokol pak musí být navržen tak, aby pouze *A* se znalostí tajemství byl schopný pravdivě odpovídat bez vyzrazení nějaké informace vedoucí k jeho tajemství. Ověřovatel *B* tedy vybere výzvu či úkol a strana *A* musí správně odpovědět. *B* pak jen kontroluje správnost odpovědi. Tento scénář se děje iteračně.

Přehled vlastností protokolu s nulovou znalostí

- Nedochází k degeneraci bezpečnosti při opakovaném použití, jelikož se stále jedná jen o málo pravděpodobnou možnost prolomení bezpečnosti.
- Nižší efektivita. Některé protokoly mají bohužel vyšší komunikační a výpočetní požadavky než klasické protokoly s veřejným klíčem.
- Neprokazatelné bezpečnostní předpoklady, podobně jako protokoly s veřejným klíčem (např. obtížnost faktorizace, problém kvadratických zbytků).

Základní verze identifikačního protokolu Fiat-Shamir

Tento protokol stojí na výzvě či úkolu *e*, který ukládá žadateli *A* dvě otázky. První otázka zjišťuje, zdali *A* uchovává tajemství *s* a druhá jednoduchá otázka, předcházející podvádění, ověřuje, zdali je *A* čestný. Pouze pravý *A* znající *s* může správně odpovědět na obě otázky. Pravděpodobnost úspěchu podvodu po položení jednoho kola je 0,5. Pro zvýšení bezpečnosti pak zvýšíme počet iterací *t* ověřování, kdy pravděpodobnost klesá na 2^{-t}.

1.4.2 Fiege-Fiat-Shamir identifikační protokol

Fiege-Fiat-Shamir (FFS) identifikační protokol poskytuje autentizaci entit na základě prokázání znalosti tajemství prostřednictvím protokolu s nulovou znalostí. Protokol tak neodhaluje žádnou část informací, vedoucí k získání tajemství strany žadatele *A*. Takový protokol pak teoreticky neodhaluje ani informace o straně *A* a může se stát vhodnou částí pro zajištění anonymní autentizace. Podobné protokoly jsou použity u el. mincí a skupinových podpisů.

Žadatel *A* prokazuje identitu ověřovateli *B* v *t* iteracích v třícestném protokolu. Schéma se dělí na 4 části:

- Zvolení systémových parametrů. Důvěryhodná strana *T* publikuje společný modulus $n = pq$ pro všechny uživatele, po výběru dvou tajných prvočísel p a q, každé shodné s 3 mod 4. Dále jsou definovány parametry k a t.
- Výběr tajemství na entitu. Každá entita *A* zvolí náhodně k-čísel $s_1,s_2,\ldots,s_k$ v rozsahu $1 \leq s_i \leq n\text{-}1$ a náhodných k-bitů $b_1,b_2,\ldots,b_k$. Vypočte $v_i=(-1)^{bi}\cdot(s_i^2)^{-1} \bmod n$ pro rozsah $1\leq i \leq k$. Entita *A* se identifikuje a registruje u *T* svůj veřejný klíč $(v_1,\ldots,v_k;n)$ a zatím si uchová tajný klíč $(s1,\ldots,s_k)$ a n. Tímto skončí jednorázové nastavení systému.

- Výměna 3 zpráv. *A* vybere náhodně *r*, $1 \leq r \leq n$-1 a náhodný bit *b* a pak vypočítá $x = (-1)^b \cdot r^2 \bmod n$ a pošle *x* (osvědčení) entitě *B*. *B* pak zašle *A* úkol, náhodný *k*-bitový vektor $(e_1,\ldots,e_k)$. Poté *A* vypočítá a pošle zpět *B* reakci: $y = r \cdot \prod_{j=1}^{k} s_j^{e_j} \bmod n$.
- *B* vypočte $z = y^2 \cdot \prod_{j=1}^{k} v_j^{e_j} \bmod n$ a ověří rovnost, že $z = \pm x$ a $z \neq 0$, kde druhá podmínka brání tomu, aby protivník uspěl volbou $r = 0$.

Bezpečnost tohoto protokolu závisí na složitosti extrakce kořenů druhé mocniny s modulem *n* pro velké celé číslo o neznámé faktorizaci. Pravděpodobnost neoprávněného přístupu je zde 2^{-kt}. Parametr *t* se nastavuje v určitém vymezeném rozsahu, jelikož pro nižší pravděpodobnost neoprávněného přístupu požadujeme velké *t* a pro spolehlivost anonymity žadatele vyžadujeme nízké *t*. Například při zvolení $kt = 20$ je pravděpodobnost neoprávněného přístupu 1:1048576. Činitel velikosti reakce *k* a počet iterací *t* pak můžou být voleny jako $k = 5$ a $t = 4$. Při nastavení poměru $k = 20$ a $t = 1$ se již nejedná o protokol s nulovou znalostí. Protokol umožňuje i jisté modifikace jako například výběr vlastního modula *n*, konverze na podpisové schéma či aplikace hašovací funkce na zprávu *z*. Více informací v [58].

Identifikační protocol GQ

Identifikační protocol GQ (Guillou-Quisquater) vycházející ze Fiege-Fiat-Shamirova schématu dovoluje redukci jak počtu výměnných zpráv, tak velikosti tajemství uživatele. Protokol opět probíhá prostřednictvím tří zpráv. Obecně je bezpečnost protokolu založena na praktické nemožnosti faktorizace velkého čísla *n*.Více informací v [58].

1.4.3 Schnorrův identifikační protokol

Schnorrův identifikační protokol je alternativou k protokolům Fiege-Fiat-Shamir a GQ. Jeho bezpečnost je založena na problému diskrétního logaritmu. Jeho návrh umožňuje snížit dobu procesu o dobu off-line výpočtů, pokud se používá jeden násobný modulus prvočísla *q*. Protokol byl navržen jako mechanismus tří zpráv. Základní princip spočívá v tvrzení znalosti tajemství *a* na straně *A*. Důkaz autentizace *A* odpovídající na výzvy *e* spočívá v asociaci tajemství *a* a veřejného klíče *v*. Tento protokol bude dále implementován v praktické části a proto je zde uveden celý matematický postup.

Výběr systémových parametrů:

- Výběr vhodného prvočísla *p* spočívá v dělitelnosti *p*-1 jiným prvočíslem *q*. Obecně se *p* vybírá v řádu $p \approx 2^{1024}$ a $q \geq 2^{160}$.
- Prvek β je vybrán z rozsahu $1 \leq \beta \leq p$-1 o multiplikativním řádu *q*.
- Každá strana obdrží autentickou kopii systémových parametrů (p,q,β) a ověřující klíč od důvěryhodné strany *T*, k ověření podpisu vydaného *T*.
- Parametr *t* se vybere z rozsahu $2^t < q$. $t \geq 40$.

Výběr parametrů na straně každého uživatele:

- Každá entita *A* má jedinečný I_A.
- *A* vybere tajný klíč *a*, $0 \leq a \leq q$-1 a vypočte $v = \beta^{-a} \bmod p$.
- *A* se identifikuje konvenčním způsobem (např. cestovní pas) straně *T* a ten ji vrátí certifikát $cert_A = (I_A, v, S_T(I_A, v))$.

Identifikační protokol pak probíhá výměnou tří zpráv v každé iteraci:

- *A* vybere náhodné *r* z $1 \leq r \leq q$-1 a vypočte osvědčení $x = \beta^r \bmod p$ a zašle *x* spolu se svým $cert_A$ k *B*.
- *B* autentizuje *A* díky ověření $cert_A$ veřejným klíčem *v* od *T* a zašle *A* výzvu, náhodné *e*, $1 \leq e \leq 2^t$.
- *A* ověří, že $1 \leq e \leq 2^t$ a vypočítá odpověď $y = ae + r \bmod q$ a pošle ji zpět *B*.

Ověření strany *A* stranou *B*:

- *B* vypočte $z = \beta^y \cdot v^e \bmod p$ a zjistí, zdali $z = x$.

Pravděpodobnost padělání je rovna 2^{-t}, čili se vybírá $q \geq 2^{2t}$. Obvykle u on-line protokolů stačí nižší *q* a u off-line aplikací se volí 2-krát větší *q* respektive *t*. Protokol ztrácí vlastnost nulové znalosti při vysoké hodnotě výzvy *e* a anonymitu pak zajišťuje obvykle jiná vrstva.

1.4.4 Slepá podpisová schémata

Slepé podpisové schéma popisuje proces, při kterém komunikují dvě strany: odesílatel *A* a podepisující *B*. Základní požadavkem schématu je zabránit podepisujícímu *B* sledování obsahu zprávy spolu s příslušným podpisem čili asociovat podepsanou zprávu s odesílatelem *A*.

Princip spočívá v zaslání od *A* k *B* určitých informací $f(z)$, které vzniknou zaslepující funkcí aplikovanou na zprávu *z*, *B* tyto informace podepíše a vrací zpět k *A*. Strana *A* pak vypočítá z podepsaných informací a pomocí odslepující funkce nový podpis, který je považován systémem za legitimní. Strana *B* tak ztratí povědomí o jeho majiteli. Mezi zaslepující funkcí *f* a odslepující funkcí *g* musí tedy platit tento vztah, $g(S_B(f(z))) = S_B(z)$, kde $f(z)$ je slepá zpráva.

Tento princip je vhodný pro stavbu systémů s elektronickými mincemi, kdy stranu *A* tvoří zákazník a stranu *B* banka. Zákazník si pak přeje anonymitu v utracení jeho peněz respektive jeho elektronického konta. Elektronickou minci pak představuje pár: zpráva *z* a podpis banky $S_B(z)$. Ve zprávě je obsažena peněžní hodnota, doba platnosti a jiné informace a podpis bankou zaručuje její integritu a původnost.
Příklad zaslepující funkce založené na RSA, viz [58]:

- klasickou konstrukcí RSA viz. část 1.2.2 získáme veřejný klíč (n,e) a soukromý klíč d,
- nechť k je určené číslo, kde platí $\gcd(n,k) = 1$,
- zaslepující funkce je definována $f(z) = z \cdot k^e \bmod n$,
- odslepující funkce je definována $g(z) = k^{-1} z \bmod n$,
- platnost podpisu S_B: $g(S_B(f(z))) = g(S_B(z \cdot k^e \bmod n)) = g(z^d \cdot k \bmod n)) = z^d \bmod n = S_B(z)$

Chaumův slepý podpisový protokol

Strana *A* obdrží podpis slepé zprávy od strany *B*, ze které vypočte korektní podpis *s* pro nezaslepenou zprávu *z*, kde $0 \leq z \leq n$-1. Strana *B* nemá znalosti o obsahu zprávy *z* a nezná ani spjatost podpisu se zprávou *z*.

Konstrukce klíčů dle RSA, kde je veřejný klíč (n,e) a soukromý klíč d. Strana *A* si vybere tajné náhodné číslo *k* z rozsahu $0 \leq k \leq n$-1 a za podmínky $\gcd(n,k) = 1$.

Protokol se dále dělí na tři části:

- Zaslepení, kde strana *A* vypočítá $z^* = z \cdot k^e \bmod n$ a pošle slepou zprávu z^* k *B*.

- Podpis, kde *B* vypočítá $s^* = (z^*)^d \bmod n$ a pošle hodnotu *s** zpět k *A*.
- Odslepení, kde *A* vypočítá korektní $s = k^{-1} s^* \bmod n$, kde *s* pak tvoří podpis *B* původní zprávy *z*.

Slepé podpisové schémata mají široký potenciál. Praktické využití lze nalézt např. v [27] či [57].

1.4.5 Nepopiratelná podpisová schémata

Na rozdíl od klasických podpisových schémat, kde ověřit podpis může kdokoliv s veřejným klíčem, u nepopiratelných (v ang. undeniable) podpisových schémat musí k ověření podpisu dojít ke spolupráci s autorem podpisu. Tato myšlenka rovněž zajišťuje určitou kontrolu podepisujícího. Schéma se uplatňuje ve dvou případech.

První případ umožňuje zákazníkovi *A*, který požaduje po poskytovateli *B* jistou službu, např. úschovu dat. Strana *B* tedy požaduje, aby strana *A* data podepsala a zaručila jejich původnost. Jestliže pak tyto podepsané data chce strana *B* později někomu ukázat jako ověřené, musí pro ověřující důkaz kontaktovat a spolupracovat se stranou *A*, které data podepsala. V takovém systému by pak strana *C* požadovala i ověření daných dat a strana *B* by musela spolupracovat s *A*. Strana *A* pak má jistotu, že *B* čestně a s vědomým *A* používá podepsaná data. Takovou kontrolu lze například využít u elektronického bankovnictví, kde zákazník *A* chce mít zaručeno, že banka *B* bude čestně disponovat s jeho aktivy.

Druhý případ je podobný. Strana *A* tentokrát prodá jisté informace straně *B*, které jsou podepsané. Jestliže se *B* rozhodne vytvořit kopii a prodat informace dál třetí straně *C*, potom *C* nemůže ověřit autentičnost informací bez strany *A*. Ovšem tento scénář nechrání od podvodného nového podepsání informací stranou *B*, jeho vlastním podpisem. Avšak pokud dojde k této situaci, tak v systému se pak snadněji vypátrá padělatel *B*.

V literatuře [58] je naznačen postup podpisového schéma Chlum-van Antwerpen, který je založen na problému diskrétního logaritmu. Existuje i distanční protokol, který má za úkol naopak zjistit přímo důkaz o padělání, více v [58]. Obecně ještě existují podpisové schémata, která umožňují získat důkaz o padělání podpisu. Tyto podpisy nevycházejí z klasických kryptografických předpokladů, ale z určité pravděpodobnosti. Více o těchto systémech v [58].

1.4.6 Metoda sdíleného tajemství

Tato část krátce pojednává o metodě sdíleného tajemství. Některé skupinové podpisové schémata, která umožňují svým jednotlivým uživatelům ve skupině zůstat v anonymitě, vyžadují tuto metodu pro zpětné odhalení jejich identity po nějaké kritické či ilegální činnosti. Metoda je v podstatě postavena na rozdělení tajemství na několik dílčích částí. Pro úspěšné získaní či použití tajemství se musí všechny dílčí části sloučit. Každá strana udržuje část tajemství a strany vzájemně spolupracují. Tajemství je tedy vyextrahováno za určitých podmínek. Metoda má rovněž i své místo v elektronickém bankovnictví, kdy se pro otevření určitého elektronického trezoru využije sdíleného tajemství. Tajemství potom zpravidla činí nějaký klíč či kód reprezentovaný číslem, který slouží k otvírání, šifrovaní, dešifrování, podepisování, ověřování a dalším činnostem.

Obecně se systémy či schémata se sdíleným tajemstvím dělí na tři mechanismy. První mechanismus počítá s tím, že pro obnovení tajemství je potřeba všech stran. Jde o prosté sdílené tajemství. Druhý mechanismus nazývající se prahové schéma vyžaduje pro získání tajemství určitý počet ze všech zúčastněných stran. Třetí typ počítá pak s jistou autorizovanou podskupinou účastníků, kteří mají možnost obnovit sdílené tajemství, které platí pro celou množinu účastníků.

Prahové schéma

Prahové schéma umožňuje obnovit společné tajemství S, již při počtu t účastníků z celkového počtu n, kde $t < n$. Princip opět spočívá v rozdělení S_i uživatelům P_i. Pokud minimální počet stran t bude chtít získat tajemství S, pak spojí své části S_i. Perfektní prahové schéma teoreticky udává, že pokud je počet uživatelů t-1 či menší, nemají ani částečnou šanci obnovit tajemství S. Funkce je bez patřičných členských částí teoreticky zcela nevyřešitelná.

Dále je požadavkem, aby se při opětovném použití obnovení tajemství nesnižovala bezpečnost. Například aby nebyly odkryty části ostatních uživatelů, které by později mohly být zneužity. Problém se řeší použitím důvěryhodného zařízení, do kterého se jednotlivé části S_i vkládají bez možnosti úniku informací o jejich hodnotách. Takové zařízení by pak mohlo mít například čtečku čipových karet, přes kterou by se jednotlivé části (uložené na jednotlivých čip-kartách) načítaly bez možnosti povšimnutí ostatních stran.

Zobecněné sdílené tajemství

Schéma poskytuje celkově množinu P uživatelů, z nichž je definována podmnožina A autorizovaných uživatelů, kde A tvoří podmnožinu P. Jakákoliv autorizovaná podmnožina entit z A, pak může obnovit tajemství S. Lze uvažovat i prahové podmnožiny o velikosti t, které mohou obnovit tajemství. Více informací o zobecnění lze nalézt v literatuře [58].

2 ANALÝZA SYSTÉMŮ S OCHRANOU SOUKROMÍ

Druhá část diplomové práce se zabývá analýzou systémů s ochranou soukromí svých uživatelů a klientů. Nejrozšířenější jsou systémy elektronické hotovosti, elektronické mince a skupinové elektronické podpisy (dále jen zkráceně: skupinové podpisy). Tyto systémy poskytují svým uživatelům anonymitu. Pokud uživatel poruší ujednaná pravidla, může jistá nadřazená autorita nebo vlastník systému odhalit jeho identitu. Analýza postupně představuje jejich základní obecné schémata, vlastnosti, jejich vývoj, typy a zhodnocení.

2.1 Elektronické mince a systémy elektronické hotovosti

Elektronické platby (v ang. E-payment) pomocí elektronických mincí (v ang. E-coin) jsou v dnešní době ve fázi návrhů a testování navržených implementací. Jedná se souhrnně o systémy elektronických hotovostí či digitálních peněz (v ang. e-cash). Dnes již existují systémy založené na principu elektronických hotovostí, které jsou ale omezeny pouze na určité aktivity, jako např. platba v hromadné dopravě nebo v uzavřených platebních systémech. Uzavřené platební systémy, které jsou postaveny na důvěryhodné straně TTP, umožňují sice pohodlné nakupování v internetových obchodech a různých aukčních portálech, ale jejich použití je neanonymní a dokonce někdy nebezpečné. Zdokonalit tyto systémy a zaručit uživateli bezpečí a soukromí je problémem dnešní kryptografie, technologie na poli marketingu, financí a síťových technologií.

Dnes se prostřednictvím Internetu provádějí jen obvyklé operace jako převody mezi bankovními konty, karetní transakce (3D Secure, standard SET) či jinými osvědčenými způsoby (platba na dobírku, příkazem, převodem či složenkou apod.). Tyto operace jsou poměrně dobře kryptograficky zabezpečeny, ale nezajišťují anonymitu.

Elektronická mince (v ang. e-coin) představuje obecnou jednotku elektronické hotovosti (e-cash), pokud neuvažujeme měnu či nominální základní hodnotu. Jedná se o určitá specifická digitální data, která jsou unikátní. Někdy se el. mince označuje jako token nebo elektronický kupón. Elektronické mince by pak na rozdíl od účtů u platebních elektronických systémů představovaly samostatnou platební jednotku, kterou by se dalo platit i za nepřítomnosti banky (bank off-line). Obecně pak můžou být schémata s elektronickými penězi (e-cash system) off-line či on-line, respektive v přítomnosti banky či bez její přítomnosti. Abychom však přenesli vlastnosti fyzické platby hotovostí do digitální podoby, je třeba uvažovat pouze scénář za nepřítomnosti banky, tedy offline schéma.

Uživatel, který by se rozhodl platit elektronickou mincí, by pak nemusel prokazovat vlastní identitu, čili by zůstal v anonymitě, ale musel by nějakým způsobem prokázat platnost mince a skutečnost že mince patří jemu. Tyto vlastnosti lze splnit například protokolem na konceptu nulové znalosti či především slepým podpisovým schématem. Jedná se pak většinou o anonymní offline systémy, které jsou rozebírány v [17], [18], [25], [26], [30], [34], [45], [54] a [63]. Většina online systémů, které nezachovávají anonymitu uživatele je poměrně dobře zabezpečena, např. digitálními podpisy, certifikáty atd. podrobně viz [37] a krátce viz [34].

2.1.1 Motivace, výhody a nevýhody

Mezi základní motivace používání elektronických peněžních systémů a e-mincí patří zaručení anonymity a ochrany soukromí uživatelů. Poskytují uživateli takové soukromí, které běžně nachází ve fyzických obchodech. Největší výhodou je zjednodušení a zrychlení internetových finančních operací (nákupy, prodeje, převody). Naopak největší nevýhodou je prostor pro nový

druh zneužití elektronických mincí, který by tato mladá technologie přinesla. Níže jsou uvedeny výhody a nevýhody konceptu elektronického peněžního systému.

Výhody:

- jednodušší nákup, prodej a převod el. hotovosti mezi uživateli na Internetu,
- téměř okamžitý převod el. hotovosti mezi uživateli a Internetovými obchody či mezi zákazníkem a poskytovatelem služby,
- zaručení anonymity čestných uživatelů,
- placení pomocí online terminálů či vlastních mobilních přístrojů v kamenných obchodech by postupně vytlačovalo tištěné peníze. Systém by ulehčil nákup zákazníkům i obsluhu prodejcům,
- možnost abstraktně centralizovat celý finanční systém na digitální podobu.

Nevýhody:

- neosvědčená a mladá technologie by mohla přinést možnost pro různé podvody,
- teoretická možnost zneužití mincí dvojitou útratou,
- nedostatek technologických a technických prostředků (velké nároky na síť),
- ztráta osobního kontaktu prodejce a zákazníka Internetových služeb,
- teoretická možnost sledování a špionáže (versus anonymita systému),
- neoprávněné stopování jedinců,
- softwarové a hardwarové selhání (ztráta elektřiny, pád programu či serveru),
- nový směr kriminality.

Shrnutím výhod a nevýhod bychom mohli dojít k závěru, že rizika převyšují výhody. Přece jenom výhody spočívají v pohodlnosti, rychlosti, zjednodušení a rozvoji obchodní činnosti. Naopak nevýhody z hlediska bezpečnosti představují větší problémy spojené se spolehlivostí dnešních technologií (software i hardware) a zaručení absolutní bezpečnosti. Zdokonalením technologií a kryptografických protokolů můžeme mnoho nevýhod minimalizovat. Nakonec zůstanou nevýhody, jako například neoprávněné stopování a možnost sledování jedinců, kde již pouze záleží na přístupu společnosti a lidském faktoru.

Problém dvojitého utracení el. mince

Elektronická mince je reprezentována pouze daty, která se mohou snadno duplikovat. Proto je nutné zabránit uživateli podruhé utratit tuto minci nějakým vhodným mechanismem. Obecně se schémata dělí na dvě skupiny: online a offline.

Při přítomnosti banky (scénář online) sama banka dohlíží na to, aby žádná mince nebyla utracena dvakrát. Obchodník pak konzultuje s bankou, zdali má el. minci přijmout. Nevýhodou tohoto schématu je neustálá interakce s bankou, která zabírá komunikační prostředky. Toto schéma je neefektivní a nepraktické. Tento scénář jako první navrhl Chaum a jeho efektivitu zkoumá článek, viz [30]. Výhodou je naopak okamžité odhalení podvodu.

Druhý scénář (offline) již ke své aktuální činnosti nepotřebuje součinnost s bankou. Obchodník akceptuje platbu el. mincí automaticky a později žádá po bance přijetí a uznání jeho utržených el. mincí. V případě dvojí útraty stejné el. mince se pak odhaluje identita podvádějícího zákazníka. Tento scénář používá například, [17], [25] a [45].

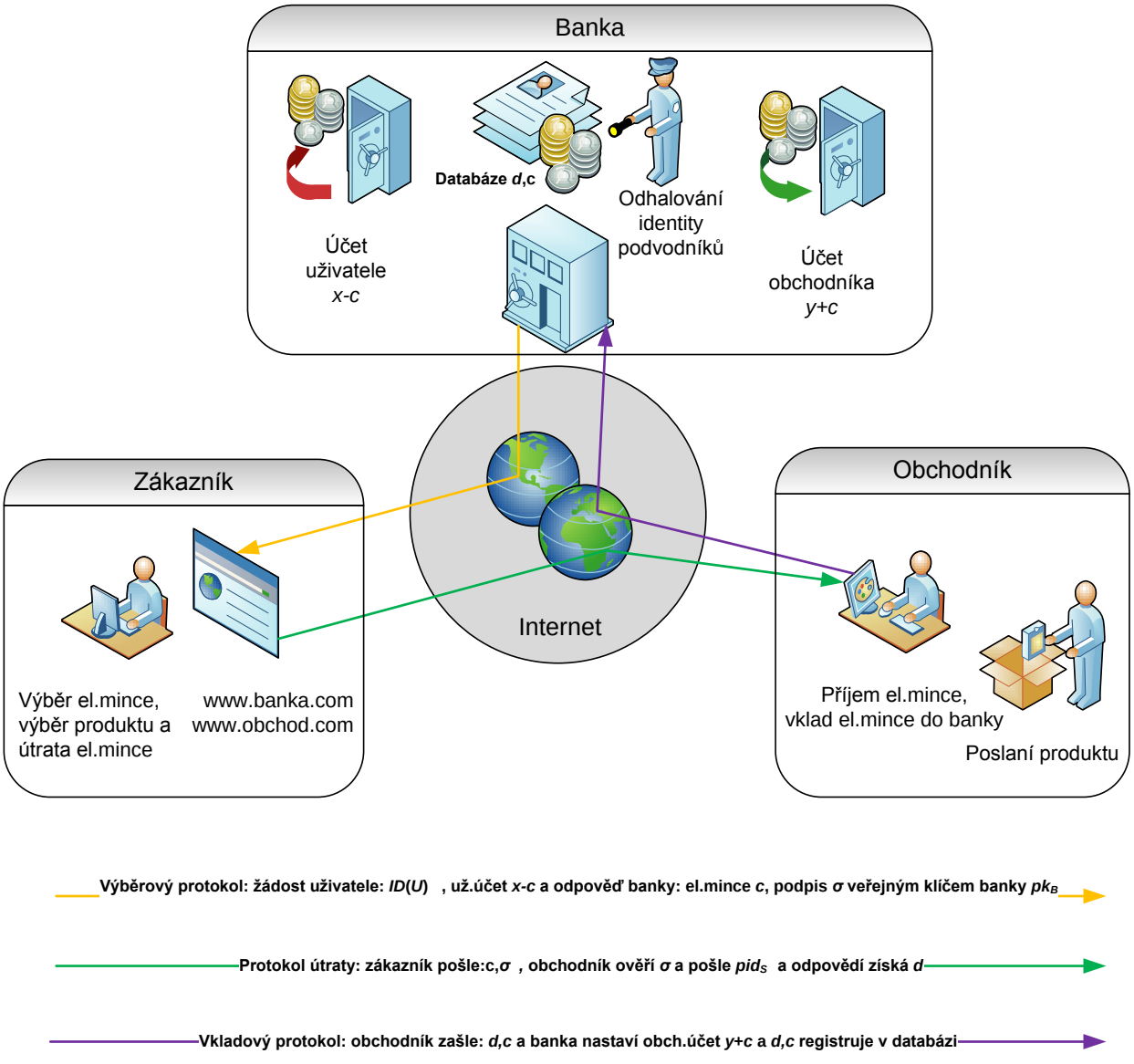
2.1.2 Obecné schéma systému elektronické hotovosti

První schéma elektronických hotovostí navrhl David Chaum na začátku 90. let. Od té doby začaly narůstat studie tohoto problému a začaly se navrhovat možná zlepšení a jiné pokročilejší přístupy. Níže uvedené obecné schéma vychází z [25] či [34].

Hlavní myšlenkou konceptu elektronické hotovosti je, že stejná strana (banka) zodpovídá za výdej (uvolnění) elektronických mincí a pozdější akceptování její hodnoty při vkladu při řádné činnosti schématu. Na Obr. 4 je zobrazeno obecné schéma systému el. hotovosti. Protokol o uvolnění mince (výběrový protokol) a protokol útraty neprozrazují žádné informace vedoucí k zjištění identity uživatele. Banka i obchodník pouze ověřují platnost přijatých mincí.

V celém schématu pak figurují 3 strany, které mohou plnit tyto funkce:

- banka, poskytovatel el. mince, ručitel el. mince,
- zákazník, uživatel, firma, která mincí platí za produkt či službu,
- obchodník, poskytovatel služeb či firma minci přijmou a vymění za produkt či službu.



Obr. 4: Obecné schéma systému elektronické hotovosti s elektronickými mincemi.

Fáze systému elektronické hotovosti:

- Vhodná konstrukce el. mince c a podpisu mince bankou σ veřejným klíčem banky pk_B.
- Výběr el. mince c z banky (výběrový protokol, v ang. withdrawal protocol), podpisu mince bankou σ s veřejným klíčem banky pk_B a získaní příslušného validačního klíče v, který závisí na $ID(U)$ uživatele, poté navíc banka odečte hodnotu mince z výše účtu uživatele U.
- Bezpečné uložení el. mince na straně uživatele.
- Útrata mince (protokol útraty, v ang. spending protocol), zákazník (uživatel) zašle obchodníkovy el. minci c a její podpis σ , který zkontroluje podpis. Pokud je mince autentická, obchodník vytvoří řetězec identifikátoru platby pid_S, který zahrnuje identitu obchodu $ID(S)$ (např. číslo účtu obchodníka) a jedinečný transakční identifikátor (náhodné velké číslo). Vložením validačního klíče v, jako tajný vstup uživatele, a pid_S, jako veřejný vstup obchodníka, vznikne vkladový klíč d (má v sobě zakódovaný pid_S), který obdrží obchodník.
- Vklad mince (vkladový protokol, v ang. deposit protocol), obchodník zasílá bance el. minci c, σ a d. Banka kontroluje podpis σ a vkladový klíč d na el. minci c. Pokud el. mince c nebyla uložena již předtím, banka vloží peníze na účet obchodníka a c a d uloží do databáze. Pokud byla mince uložena již předtím, databáze obsahuje c,d'. Jestliže identifikátor platby pid_S dekódovaný pomocí d a identifikátor pid_S' dekódovaný databázovým d, se rovnají $pid_S = pid_S'$, pak je jednoduše nový vklad c, d odmítnut. Jestliže však $pid_S \neq pid_S'$, pak systém umí efektivně zkonstruovat z dat c, d a d' původní identitu uživatele $ID(U)$, jelikož je zřejmé, že uživatel platil stejnou mincí podruhé. Záleží pak na podmínkách systému, zdali přičte obchodníkovi peníze na účet.

Bezpečnostní a přídavné protokoly:

- stopování podvodníků a odhalení jejich identity,
- zmrazení el. mincí podvodníkům,
- rozměnění el. mincí,
- zrušení elektronické mince.

Záruky, ochrany a vlastnosti systému pro banku:

- nemožnost padělání mince, pro případ kdy se spikne obchodník se zákazníkem, musí vždy existovat unikátní el. mince, která vzniká při konstrukci,
- zachycení uživatelů s dvojitým utracením el. mince.

Záruky, ochrany a vlastnosti systému pro zákazníka:

- anonymita uživatelů,
- nemožnost falešného obvinění,
- integrita hodnot elektronických mincí,
- záruku obsluhy korektního (čestného) zákazníka čestným obchodníkem,
- bezpečný výběr a uložení el. mince z banky.

Záruky, ochrany a vlastnosti systému pro obchodníka:

- integrita autentizace hodnot elektronických mincí,
- bezpečný protokol útraty,

- bezpečný vklad el. mincí do banky,
- odstranění možnosti padělání el. mince.

2.1.3 Kryptografické prvky v systémech elektronické hotovosti

Podle obecného schématu systému el. hotovosti lze systém rozdělit na tři základní protokoly (výběr, útrata a vklad). K celkové funkčnosti se ještě přidávají protokoly odhalení podvodníka, rozměnění mincí atd. Pro každý protokol lze nalézt vhodnou kryptografickou metodu, která splňuje požadavky protokolu a bezpečnostní záruky pro zúčastněné strany. Používají se slepá podpisová schémata, RSA, protokoly identifikace s nulovou znalostí, metody sdíleného tajemství atd. Tyto mechanismy jsou popsány v kapitole 1.

Konstrukce

V první fázi dochází ke konstrukci klíčů, parametrů a databází systému. Parametry a klíče se vytváří na základě použití kryptografických primitiv, která později určují účinnost a bezpečnost všech následujících protokolů.

První starší systémy, např. viz [45], používaly kryptosystém RSA, kdy se vytvořil veřejný modulus n, jehož faktorizaci (tajný klíč) znala pouze banka a dále se určila bezkolizní jednocestná funkce f a bezpečnostní parametr k. Na základě těchto parametrů pak proběhl protokol výběru. Některé novější a efektivnější systémy jsou také založeny na RSA s kombinací speciálních podpisových schémat, viz [18], [25].

Systémy založeny na problému diskrétního logaritmu např. [17], [25], [26], [30], [63] při konstrukci vytváří náhodný generátor g z grupy G o řádu q a zbylé tajné a veřejné parametry banky a uživatele, které záleží již na konkrétním typu.

Existují také systémy založené na eliptických křivkách či v kombinaci s bilineárním mapováním viz [25], [26] a [30].

Protokol výběru

Jde o komunikaci mezi bankou a uživatelem při inicializaci mince. Banka potřebuje vědět *ID* uživatele, aby mu mohla uvolnit mince z účtu. Po tomto kroku nyní uživatel požaduje nesledovatelné mince. Tato podmínka platí v případě čestného jednání uživatele. Banka musí vytvořit el. minci takovým způsobem, aby pravděpodobnost padělání byla zanedbatelná či téměř nulová (padělám se myslí validnost duplikátu již utracené mince).

První krok je pro většinu schémat podobný, kdy se uživatel musí identifikovat bance kvůli jeho účtu, pomocí potvrzení znalosti uživatelova tajného klíče, viz klasické podpisové schéma.

Ve druhém kroku starší systémy, např. viz [45], používaly slepé podpisové schéma, které zaručuje, že uživatel získá podepsanou minci c, např. Schnorrovo slepé podpisové schéma. Tento způsob byl ale pro banku nevýhodný, jelikož přesně nevěděla zdali je podepisovaná el. mince v korektní formě. Další možností bylo použití techniky prokázání nulové znalosti (mechanismem cut-and-choose), viz [63] a [54]. Ale tento způsob je poměrně neefektivní, protože velikost dat el. mincí byla velká.

Efektivním řešením je použití tzv. závazkového schéma (v ang. commitment scheme), což je také v podstatě slepé schéma, které spočívá v tom, že uživatel zná tajemství w_1 a náhodné číslo w_2 a tyto dvě hodnoty spolu sváže (v ang. commit). Takový závazek, který ukryje tajemství w_1, pak má podobu $h = \text{commit}(w_1,w_2) = g_1^{w_1} g_2^{w_2}$. Generátory g_1 a g_2 jsou veřejně známé. Celý mechanismus souvisí s tvrzením, že nelze efektivně vypočítat x z $g_1 = g_2^x$, ale pokud máme dvě stejné mince $(w_1, w_2) = (w'_1, w'_2)$, tak lze snadno vypočítat skrytou identitu uživatele. S tímto způsobem přichází [17].

Uživatel tento závazek (obsahující tajné číslo, tajný podpis uživatele a dva náhodné prvky) pošle bance. Banka pak vygeneruje nové náhodné číslo (jeden z prvků v závazku), které pošle zpět uživateli. Uživatel přepočítá nový závazek vzhledem k přijatému náhodnému číslu. Podobně banka svým způsobem přepočítá závazek, který nyní podepíše a podpis předá uživateli, více viz [25] a [34]. Po skončení protokolu uživatel vlastní validační klíč (com,$h(z)$), tajný klíč (w_1,w_2) pro jednorázový podpisový klíč a jednorázový ověřovací klíč a slepý podpis mince vydaný bankou.

Obecně použití závazku (ang. commitment) spočívá ve svázání (v ang. binding) a skrytí (v ang. hiding) určitých hodnot. Ve skutečnosti je nemožné zajistit bezpodmínečně bezpečné obě funkce v jednom závazku najednou. První funkce závazku tj. svázání či vazba určuje, že informace v závazku již nemůže podepisující později změnit. Druhá funkce skrytí znamená, že ověřovatel nemůže teoreticky zjistit co je uvnitř závazku a může při určité pravděpodobnosti pouze hádat, více informací v [35].

Jedno z prvních schémat bylo uvedeno v [61], tzv. Pedersenovo závazkové schéma. Pedersen navrhl schéma na základě problému diskrétního logaritmu. Z problému RSA vychází Fujisaki-Okamotovo závazkové schéma, viz [39]. Pedersenovo závazkové schéma je využito u elektronických hotovostních systémů [25], [30], [26] a [34].

Protokol útraty

Komunikace mezi uživatelem a obchodníkem. Uživatel chce zůstat v anonymitě a obchodník potřebuje mít jistotu o platnosti el. mince, kterou vlastní uživatel.

Ve starších schématech [45] uživatel zaslal minci obchodníkovi a jednoduše odpovídal dle náhodného binárního řetězce vygenerovaného obchodníkem. Obchodník pak pouze ověřil správnost odpovědí, které pak uchoval jako důkaz o platbě pro banku. Rovněž v [63] se používá neinteraktivní důkaz nulové znalosti. Tentokrát je přidán na validní haš řetězec dotazů i čas a data transakce.

V novějším schématu [34] obchodník posílá identifikátor platby *pid* uživateli, který extrahuje údaje o platbě, tento identifikátor musí být jedinečný (kombinace náhodného čísla, pořadového čísla transakce, časového razítka a jiných údajů). Poté uživatel vypočte pomocí tajného podpisového klíče jednorázový podpis *pid*. Poté posílá slepý podpis mince bankou, validační klíč a nově vypočtený jednorázový podpis *pid*. Obchod pak ověřuje ověřovacím klíčem banky slepý podpis el. mince bankou, který musí být spjatý spolu s veřejným klíčem uživatele. Rovněž musí být korektní i ověření jednorázového podpisu *pid*. Pokud je vše validní, pak obchodník pošle zboží a ukládá si veřejný klíč uživatele, slepý podpis el. mince bankou, *pid* a jeho jednorázový podpis.

Protokol vkladu

Komunikace mezi obchodníkem a bankou. Obchodník chce uložit el. minci do banky, která mu přičte její hodnotu na konto.

Ve všech schématech obchodník jednoduše bance předkládá výstupní hodnoty protokolu útraty. Z kryptografického hlediska se v tomto protokolu řeší pouze porovnání, zdali el. mince spolu s identifikačními údaji o transakci nebyly vkládány již někdy předtím. V případě že el. minci ještě nepřijímal, akceptuje žádost, připíše částku hotovosti na konto obchodníka a hodnoty o el. minci a transakci si ukládá do databáze. Pokud banka najde ve své databázi již uložené shodné parametry o transakci, žádost zamítne. Pokud nalezne stejné el. mince (stejné sériové čísla) s odlišnými transakčními údaji, podnikne kroky k nalezení identity uživatele dle příslušného algoritmu.

Protokol odhalení identity podvodníka

Starší systémy umožňovaly, aby banka uvolnila minci tak, že neměla pak informace o tom, jak se mince utratila a zákazník mohl anonymně minci utratit. Pokud zákazník minci utratí po druhé, banka zpětně vyhledá ID podvodníka, který je spjatý s el. mincí v její databázi, viz [45]. Tento způsob ale požadoval čestnou banku nebo využití důvěryhodné třetí strany. Jelikož pro tyto strany čestný uživatel nezůstával anonymní, začalo se uvažovat o jiných způsobech odhalení.

Některé schémata používají k odhalováním podvodníků pomoc agentů, důvěryhodných stran TTP, které mají příslušné práva a možnosti, viz [25]. Agenti drží ve vlastnictví nějaký tajný revokační klíč, kterým při porušení pravidel mohou odhalit identitu podvodníka. Jiná schémata [17], [25], [26] či [34], kde banka či kdokoliv umí odkrýt identitu podvodníka přímo výpočtem, vycházejí z neoprávněného dvojího utracení mince. Při držení dvou mincí o stejných sériových číslech může banka, obchodník či kdokoliv v systému efektivně vyextrahovat tajnou hodnotu, která vede k určení identity majitele příslušných elektronických mincí. Požadavkem však je, aby případný útočník neměl nijak ulehčenou situaci při extrahování tajné hodnoty.

2.1.4 Příklad systému elektronické hotovosti: Brandovo schéma

V této části je uvedeno Brandovo schéma elektronické hotovosti (Brand's eCash scheme). Z tohoto schématu nepřímo vychází teoretický návrh autentizačního systému, který bude implementován v praktické části.

Brandovo schéma popisované v [17] a [18] je založeno na použití verifikačního klíče *vk* pro jednorázové podpisy, které získá uživatel při slepém výběru el. mince z banky. Tajný klíč *sk* slouží k validaci el. mince a je zkonstruován na základě identity uživatele, která je tak utajena. Ověření identity uživatele banka realizuje pomocí *vk*. Pro ověření mince, zde tedy $c = vk$, banka uvolňuje vk_B a tajným klíčem sk_B, kterým podepisuje el. minci (veřejný klíč) $\sigma = \mathrm{sig}_{skB}(vk)$, podrobněji v [17], [18] [34]. První tři části schématu mezi uživatelem a bankou jsou zobrazeny na Obr. 5.

Konstrukce:

- pomocí závazkového schématu, viz [34] či [61] se nastaví parametry (G_q, generátory g_1 a g_2),
- nikdo v systému neumí vypočítat $\log_{g_2}(g_1)$,
- banka sejme uniformně náhodné $G \in G_q$ a $w \in \mathbb{Z}_q$ a vypočte veřejné $H = G^w$,
- tzv. svědectví w zůstane v tajnosti jako tajný klíč sk_B a hodnoty (G,H) se uveřejní jako klíč vk_B.

Registrace uživatele:

- každý anonymní uživatel je reprezentován tajnou identitou $g_U = g_1^U g_2$,
- hodnota U je náhodná hodnota uniformně volena uživatelem při registraci do systému, banka dostane pouze g_U,
- po registraci banka vypočte $h_U = g_U^w$ a pošle h_U uživateli, který tedy vlastní g_U, h_U a U a banka vytvoří záznam, že příslušný uživatel je registrovaný pod veřejným identifikátorem g_U.

Protokol výběru:

- pomocí svědectví w se ověří žádost $X = (H, G, g_U, h_U)$, kde $H = G^w$ a $h_U = g_U^w$,
- při slepém podpisovém schématu uživatel vybere uniformně náhodné $s \in \mathbb{Z}_q$ a vypočítá podpis $g_U^s = (g_1^U g_2)^s = g_1^{Us} g_2^s$,
- uživatel spočítá $com = g_U^s$ a $w_1 = Us \bmod q$ a $w_2 = s$, pak platí rovnost $com = g_1^{w_1} g_2^{w_2}$, kde w_1 a w_2 jsou jistá svědectví,
- uživatelův důkaz o znalosti spočívá ve znalosti w_1 a w_2 a výpočtu zprávy a,
- jistá zpráva $z = a$ a $com = g_U^s$ jsou části zhašované zprávy,
- uživatel obdrží slepý podpis bankou $\sigma_B = \mathrm{sig}_{skB}(vk)$, kde $vk = (com, a)$ a vlastní tajný $sk = (w_1, w_2)$, klíče vk a sk pak použije pro jednorázové podpisy,
- banka na základě a a g_U sníží hodnotu účtu.

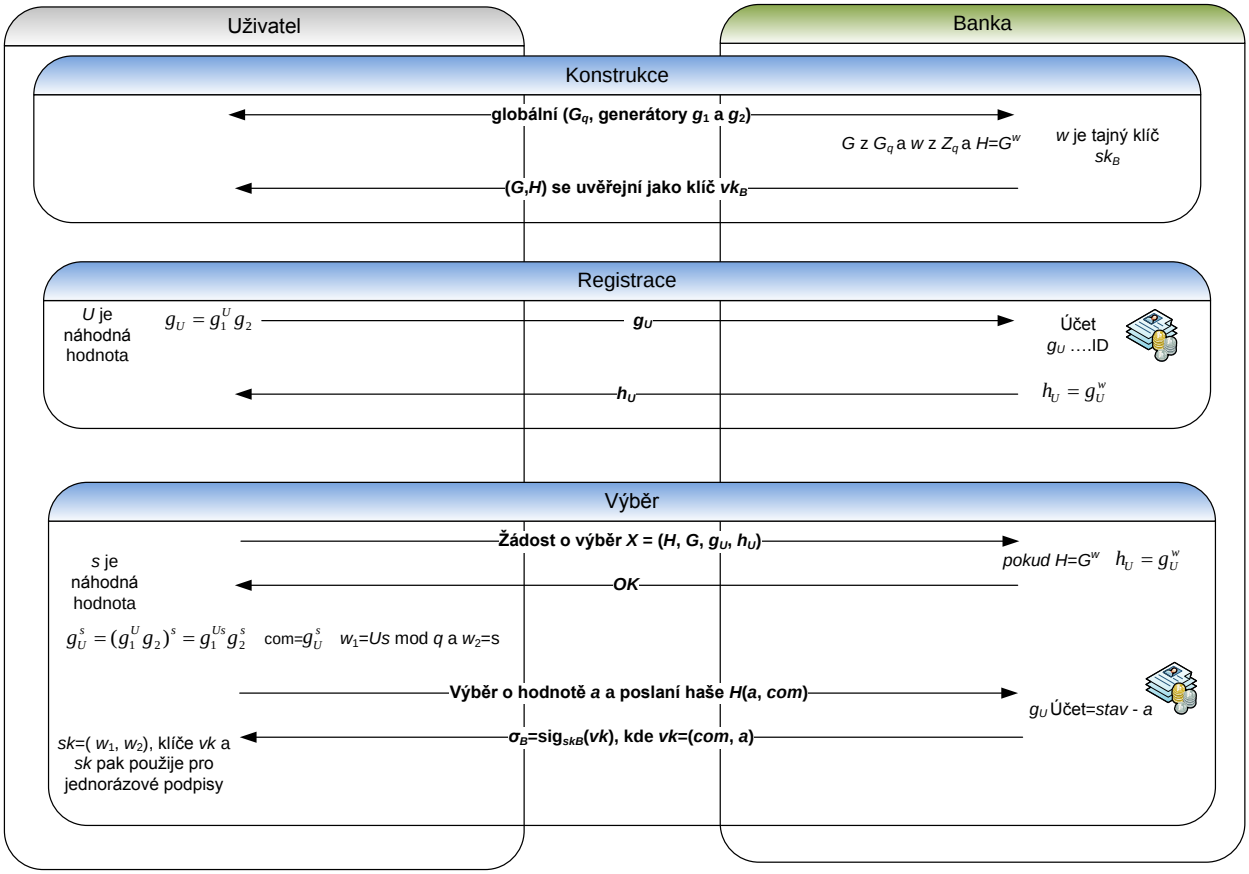
Protokol útraty:

- obchod nastaví *pid*, zahrnující identitu obchodu, unikátní id transakce, časové razítko či počítadlo a pošle ho uživateli,
- uživatel vypočítá jednorázovým podpisem $\sigma = sig_{sk}^{OT}(pid)$ a vrací obchodu (vk, σ_B, σ),
- obchod ověřuje fcí $ver_{vkB}(vk, \sigma_B)$ = validní a $ver_{vk}^{OT}(pid, \sigma)$ = validní,
- obchod odbaví zakázku a uchová si $(vk, \sigma_B, pid, \sigma)$.

Protokol vkladu:

- obchod posílá (třeba i několik) požadavků bance ve tvaru $(vk, \sigma_B, pid, \sigma)$,
- banka kontroluje *pid* a dekóduje identitu obchodu, v případě že již má v databázi uložené *pid* tak vklad zamítá,
- banka rovněž ověřuje, že $ver_{vkB}(vk, \sigma_B)$ = validní a $ver_{vk}^{OT}(pid, \sigma)$ = validní a uchová si $(vk, \sigma_B, pid, \sigma)$.

Bezpečnostní opatření banky proti podvodníkům, kteří utratí minci víc než jednou, spočívá v možnosti výpočtu tajné identity uživatele U. Pokud má banka dvě mince se stejným sériovým číslem, potom může počítat jejich w_1, w_2 a w_1', w_2' dle $com = g_1^{w_1} g_2^{w_2}$. Na základě vztahu $w_1'(w_2')^{-1} \bmod q = w_1(w_2)^{-1} \bmod q = Us(s)^{-1} \bmod q = U$, podrobněji v [17], [18], [34].



Obr. 5: Komunikace mezi uživatelem a bankou u Brandova schématu el. hotovosti.

2.1.5 Zhodnocení systémů elektronické hotovosti a elektronických mincí

Systémy a schémata el. mincí či el. hotovostí se postupně vyvíjely, dlouhou dobu zdokonalovaly a stále se budou zlepšovat. Jednotlivé protokoly daných schémat (výběr, útrata a vklad) mohou mít různé výpočetní náročnosti a různé úrovně zabezpečení. Starší návrhy byly většinou málo efektivní a pro rozsáhlejší systémy zcela nepraktické. Dále také některé starší schémata obsahovaly různé implantační nedostatky a bezpečnostní slabiny. Novější návrhy viz např. [25], [26], [30] či [34] nabídly efektivnější řešení. Například schéma [25] zefektivnilo výběrový protokol na výpočetní složitost O(l+k) ze starších návrhů [45], [63], kde byla složitost O(2^l*k), kde l značí počet el. mincí a k bezpečnostní parametr.

Rovněž se měnily i bezpečnostní primitiva jednotlivých protokolů. U starších návrhů se využíval u výběrového protokolu koncept nulové znalosti. To přineslo zanedbatelnou pravděpodobnost uhodnutí tajných parametrů, např. 2^{-40}. Novější návrhy [25], [30] a [34] přecházely na využívání závazkových protokolů ([61], [39]), které jsou založeny problému diskrétního logaritmu či problému RSA a zlepšují tak efektivitu i bezpečnost.

Dále se vyskytlo několik návrhů systémů el. hotovosti za použití skupinových slepých podpisů, např. [57], kde se ve scénáři vyskytuje centrální banka, která řídí členy a kontroluje jejich činnost. Ostatní banky, které jsou členy ve skupině pod centrální bankou, pak mohou vydávat el. mince. Obchodník potom neví, ze které banky byla daná mince uvolněna a zákazník tak získává novou vrstvu anonymity. Schéma naopak postrádá některé funkce jako je dělitelnost el. mincí a přenosnost a rovněž je navržen jako online systém. Další schéma [54] počítá

s nějakou důvěryhodnou stranou jako se skupinovým manažerem a zákazníci tvoří členy skupiny.

Systémy jsou teoreticky téměř připraveny, jejich současné známé koncepty jsou bezpečné na modelu random oracle, např. [25], [26], [30]. Random oracle se považuje za tzv. most mezi teoretickou a praktickou kryptografií, více o RO v [9]. V současnosti se vyvíjí a odlaďují koncepty, které budou bezpečné v reálném standardním modelu.

Nicméně po praktické stránce se objevuje několik problémů a překážek stojících před implementací systému el. hotovosti do reálné praxe. Bankovnictví má nyní zavedené pevné systémy, které jsou léta funkční a bezpečnostně vyladěny a rovněž se neprojevuje přílišný zájem o anonymitu ze strany zákazníků. Dalším zádrhelem je vybudovaní důvěryhodných třetích stran a arbitrů, kteří by stíhali a trestali podvodníky a ospravedlňovali čestné uživatele, což by nebylo levné pro finanční poskytovatele. Dalším již zmíněným problémem stále zůstává celková bezpečnost systému el. hotovosti, která se doposud pouze teoreticky testuje v RO modelu. A v neposlední řadě je kladen požadavek na dostatek síťových a technických prostředků na straně bank a obchodníků, kteří by museli spolehlivě zpracovávat všechny transakce, které by vzhledem ke kryptografickému zabezpečení a velkému počtu el. mincí vyžadovaly velké nároky na výpočetní výkon a paměťové prostředky.

2.2 Skupinové elektronické podpisy

Skupinové digitální podpisy (v ang. group digital signatures), poprvé představeny v [46], dovolují jakémukoliv členu skupiny podepsat digitální data jménem skupiny, tedy za skupinu. Ověřující entita respektuje jeden veřejný klíč celé skupiny a identita entity, která podpis vytvoří, zůstává utajena. Podepisující člen tedy zůstává anonymní, ale jeho identita může být za určitých podmínek vystopována a odhalena určitou autoritou. Každá skupina obsahuje několik členů, kteří jsou anonymní a mohou podepisovat data pomocí tajného klíče. Každý podpis je pak ověřen pomocí veřejného skupinového klíče.

Do skupiny přidává členy skupinový manažer, který může odhalit identitu člena. Někdy se funkce skupinového manažera (group manager) rozděluje na dvě entity: skupinového (náborového) manažera (v ang. adding/joining manager) a odhalujícího (revokačního) manažera (v ang. revocation manager). Odhalujícího manažera také někdy zastupuje důvěryhodná třetí strana TTP či nějaká otevřená autorita OA.

Obecně v dnešní době existuje celá řada typů a návrhů skupinových schémat. Schémata se na rozdíl od původních a starších návrhů [46], [47] různě zaměřují na určité vlastnosti. Některé schémata [7], [11], [22], [42], [48] či [55] se pokouší zaručit naprostou anonymitu, tzv. odvolatelnost není většinou založena na jednom prvku, ale např. na souhře části skupiny. Některé kladou požadavky na záruky bezpečnosti [1], [14], [15], [36], [53], další na krátkou délku podpisu [13], [38], [72] a jiné na efektivitu [15], [16], [21], [23], [40], [50], [51], [53] či [62]. Postupem času se návrhy vylepšovaly a zvyšovala se jak bezpečnost, tak i efektivita. Dnešní návrhy se snaží nastavit co nejefektivnější schéma, které zaručuje potřebnou bezpečnostní hladinu, při co nejkratším podpisu a zaručené anonymitě.

2.2.1 Motivace, výhody a nevýhody

Jedním z důvodů, proč se začaly vyvíjet skupinové podpisy, bylo nějakým způsobem zachovat anonymitu podepisujících členů v případech, kdy není jejich identita pro ověřovací stranu nutná. Na druhou stranu ověřovatel požaduje záruku bezpečnosti a jistoty, že jednají s jistou stranou a ne s podvodníkem.

Obyčejné digitální podpisy, které jsou bez identity, resp. použití certifikátu, také anonymní, by tento požadavek nesplnily. Takovýto model je výhodný např. pro zaměstnance velkých firem, kteří podepisují dokumenty za firmu a přitom neodhalují vlastní soukromé informace. Rovněž se tak snižuje počet veřejných klíčů na jeden skupinový veřejný klíč.

Dále se skupinové podpisy mohou uplatnit při anonymních přístupových [5] či pověřovacích [7], [29] systémech (v ang. anonymous access/credentials system). Uživatel je evidován v rámci určité skupiny. Při použití služby, přístupu či dat se zaručuje pouze znalostí (podpisem), že patří do jisté skupiny. Uživatel tak může anonymně a nestopovatelně využívat některé online služby a jeho anonymita se odhaluje v případě porušení pravidel.

Jako nevýhoda se jeví větší délka podpisu, kdy obyčejné podpisové schémata mají délku podpisu nejkratší, podpisové schéma zahrnující identitu přibližně dvojnásobnou a nakonec skupinové schéma několika násobně větší, kdy většina starších schémat měla délku podpisu závislou na velikosti skupiny. Tato nevýhoda je nežádoucí například u systému VSC (Vehicle Safety Communications), viz [13], kde automobily komunikují zabezpečeně mezi sebou při jízdě po silnici a navzájem se upozorňují na brzdění, změnu pruhu a jiné užitečné informace, které pomáhají řidiči. Zde pro nejrychlejší odezvu požadujeme co nejkratší podpis a rychlé ověření podpisu.

Výhody:

- anonymita a ochrana soukromí člena skupiny,
- bezpečnost je adekvátní k běžným digitálním podpisům,
- škálovatelnost ve firemní hierarchii,
- jeden ověřovací klíč na celou skupinu.

Nevýhody:

- velká délka podpisu,
- výpočetní náročnost,
- možnost různých nových zneužití.

2.2.2 Obecné fáze skupinových podpisů

Konstrukce

První proběhne konstrukce a inicializace veřejného skupinového klíče *PKg* a soukromých klíčů *SKu*. Také proběhne i konstrukce různých proměnných, které se liší podle použitých kryptografických primitiv a inicializace databází. Zde dochází k podstatnému rozdílu mezi statickými skupinovými podpisy a dynamickými skupinovými podpisy. U dynamických se vytvoří veřejný klíč skupiny, nastaví se konstrukční prvky a poté se skupina členů může libovolně rozrůstat či zmenšovat. U statických schémat se veřejný klíč a tedy celá konstrukce odvíjela od počtu členů, kde nové nastavení probíhá po každé změně počtu členů. Tyto návrhy schémat byly neefektivní.

Připojení/vydaní/zápis

Pokud se jedná o statické schéma, dochází k tomuto kroku již při konstrukci, kdy každý člen obdrží vlastní odlišný soukromý podpisový klíč *SKu*. V jednodušším případě tento klíč koresponduje s údaji o identifikaci člena, které jsou uloženy v databázi u manažera skupiny.

V pokročilejších případech u dynamických schémat každý nový člen získá soukromý klíč *SKu* od skupinového manažera bez potřeby přepočítávat a ustanovovat nové klíče pro ostatní členy a nový veřejný skupinový klíč. Dále záleží na způsobu zachování anonymity.

Ve většině schémat obdrží při výdeji tajného klíče revokační (popř. skupinový) manažer či TTP/OA identifikační údaje *IDu* o členovi. Takovéto schémata mají třetí stranu, která má neomezenou moc. Existují ale taková schémata, které se snaží zachovat plnou anonymitu členů. Zde jsou *IDu* posílány revokačnímu manažeru v zašifrované formě (závazkové schéma) a ten jejich identitu může odkrýt až při splnění určitých podmínek nebo na pokyn nadřízené oprávněné kontroly. Nejčastěji se jedná o zdvojený podpis. Existují i takové typy schémat, kdy pro odkrytí identity je potřeba několik členů, či různá souhra entit (prahové skup. podpisy, schémata se sdíleným tajemstvím). Tyto schémata se někdy nazývají jako demokratické skupinové podpisy.

Podpis dat

Při elektronickém podpisu dat (zprávy *z*) každý člen ve skupině použije svůj unikátní soukromý klíč *SKu*, který koresponduje se společným veřejným skupinovým klíčem *PKg*. Zprávu *z* spolu s podpisem σ = Podpis(*z, SKu*) odesílá druhé straně.

Ověření podpisu

Entita, která obdrží podpis σ spolu se zprávou *z*, může pomocí veřejného klíče *PKg* ověřit autentičnost a skupinovou původnost zprávy. Klíč *PKg* získá při příjmu zprávy a podpisu, nebo od TTP či skupinového manažera. Ověřovatel pak zjišťuje, zdali je podpis validní na základě algoritmu Ověření (σ, *z, PKg*).

Stopovaní a odhalení

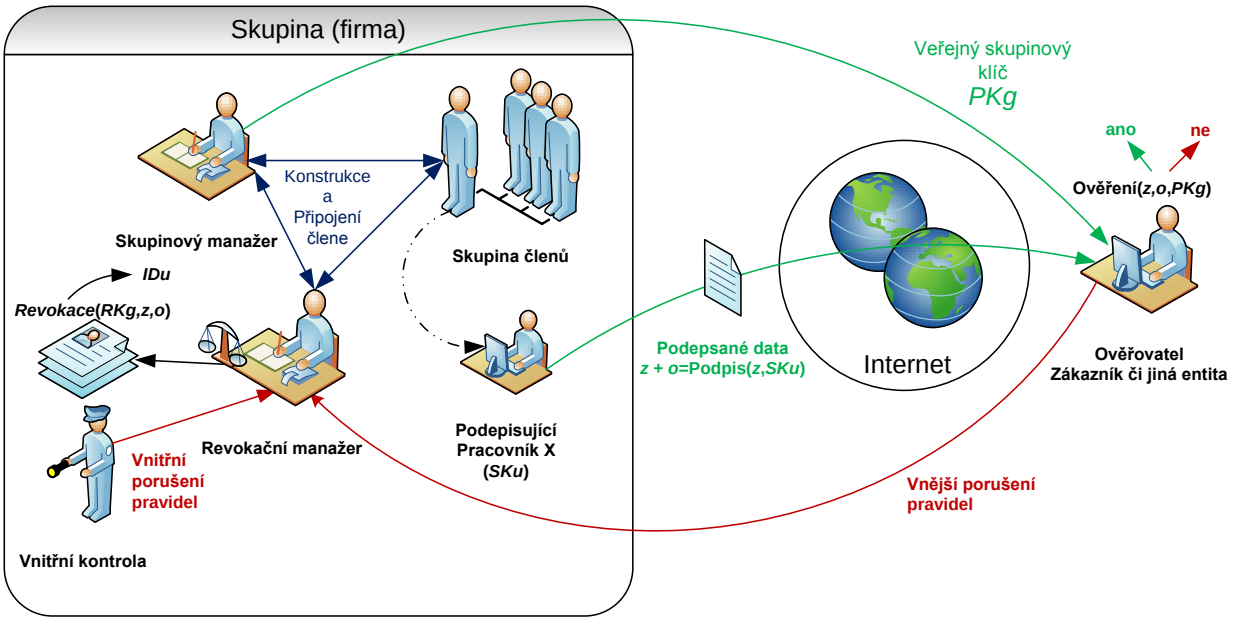
V případě, kdy nějaký člen poruší nějaké pravidlo, nebo je žádoucí zjistit, kdo podepsal určité data, je možné odkrýt identitu příslušného autora podpisu. Revokační manažer potřebuje pouze originální zprávu, podpis a vlastní tajný revokační klíč *RKg*, dle kterého odhalí tajný klíč uživatele a tak jeho identitu. Je zřejmé, že pro klasické firemní skupinové podpisy je vhodný model s revokačním manažerem, který může odkrýt identitu kdykoli je to potřeba. U modelu el. mince je naopak žádoucí možnost odkrýt identitu až po dvojím utracení el. mince, viz např. [71].

2.2.3 Obecné schéma skupinových podpisů

Schémata skupinových podpisů se mezi sebou mírně odlišují a to jak z hlediska funkcí jednotlivých entit, tak i z hlediska použitých protokolů a primitiv. Níže je uvedeno zobecněné schéma skupinových podpisů, které je pro přehlednost odlehčeno od kryptografických primitiv či protokolů. Na Obr. 6 je znázorněno obecné schéma skupinových podpisů.

Entity:

- skupinový manažer (správa a konstrukce klíčů, přidává nové členy, odstraňuje členy),
- člen skupiny (anonymně podepisuje data pomocí svého tajného klíče),
- revokační manažer, někdy TTP, OA (stopuje a odhaluje identitu členů),
- venkovní entity, ověřovatelé (ověřují podpisy pomocí veřejného skupinového klíče).



Obr. 6: Grafické vyjádření obecného schéma skupinových elektronických podpisů.

Základní vlastnosti

- Nepadělatelnost (v ang. Unforgeability): Pouze člen skupiny může platně podepsat data jménem skupiny a je vysoce nepravděpodobné, aby někdo jiný vytvořil validní podpis.
- Anonymita (v ang. Anonymity): Pokud máme validní podpis nějaké zprávy, lze jen těžko odhalit identitu. Je to prakticky výpočetně neřešitelné.
- Úplná anonymita (v ang. Full-Anonymity): Pokud útočník má validní podpis nějaké zprávy a všechny tajné klíče všech členů skupiny, pak lze jen těžko odhalit identitu podepisujícího. Je to prakticky výpočetně neřešitelné.
- Nesledovatelnost (v ang. Unlinkability): Je těžké, prakticky výpočetně neřešitelné, rozhodnout o dvou podepsaných zprávách, že pocházejí od jednoho podepisujícího.
- Nezastupitelnost (v ang. Exculpability): Nikdo ze skupiny, ani skupinový či revokační manažer, nemůže podepsat data za jiného člena. Možnost padělání podpisu je vysoce nepravděpodobná.
- Stopovatelnost (v ang. Traceability): Revokační manažer (pokud není, tak skupinový manažer) má vždy možnost otevřít validní podpis a ověřit identitu aktuálního podpisu.
- Úplná stopovatelnost (v ang. Full-Traceability): Pokud se nějak skupinový manažer domluví s členem skupiny, tak nemohou spolu vytvořit nevystopovatelný validní podpis.
- Odolnost proti domluvě (v ang. Coalition-resistance): Množina členů nemůže vydat validní podpis za někoho jiného ve skupině, při spojení jejich znalostí.
- Správnost (v ang. Correctness): Všechny podpisy produkované jakýmkoliv členem při správném podpisu musí být akceptované při ověření.
- Odolnost proti zneužití (v ang. NoFraming): Jestliže se revokační manažer, skupinový manažer a nějaký skupinový člen domluví, nemohou podepsat data za nevyžádaného člena ve skupině.
- Okamžité odvolání (v ang. Immediate-revocation): Je nemožné pro člena skupiny odvolat v čase t vygenerovaný validní podpis. Toto slouží pro snížení sporů při asynchronní komunikaci, více viz [36].

- Únik na svobodu (v ang. Leak-freedom): Je nemožné pro podepisujícího dokázat vlastnictví daného skupinového podpisu bez odevzdání svého privátního klíče, i když jej vlastní všichni ostatní členové, skupinovému manažeru, více viz [36].

2.2.4 Porovnání schémat skupinových podpisů

V této části budou porovnána vybraná schémata skupinových podpisů. Porovnává se zde především efektivita, délka podpisů, délky klíčů a rozsahy použití. Srovnávají se použitá kryptografická primitiva a problémy na kterých stojí bezpečnost jednotlivých schémat. V dnešní dostupné literatuře lze nalézt pouze srovnání pro malý počet schémat. Srovnání bývá jen z určitých pohledů. Například v [38] lze nalézt konkrétní porovnání rychlosti ověřování el. podpisů mezi anonymními, skupinovými, kruhovými schématy a klasickým schématem založeném na identifikaci. Avšak je zde zastoupeno pouze jedno skupinové podpisové schéma. V Tab. 1 je porovnáváno několik vybraných schémat skupinových el. podpisů.

První sloupcová položka udává autory daného schématu, rok vydání či publikace (nemusí přesně korespondovat s rokem v literatuře) a odkaz na zdroj v literatuře.

Druhý sloupec obecně klasifikuje typ skupinového podpisu a jeho funkční odlišnosti, popřípadě vhodnost použití. Například pro statické skupiny (vhodné pro ucelené skupiny) či pro dynamické skupiny (členové se můžou přidávat a vylučovat ze skupiny bez vlivu na hodnoty klíčů a jiné parametry ve schématu).

Ve třetím sloupci budou uvedena kryptografická primitiva, na kterých je návrh schématu postaven. Pro větší přehlednost jsou uváděny pouze zkratky, např. SRSA označuje silné RSA (v ang. strong-RSA), SDH označuje silný Diffie-Hellman protokol, další zkratky viz seznam použitých zkratek. Funkčnost jednotlivých primitiv a funkční význam jednotlivých zkratek bývá definována pak v příslušném schématu, viz odkazy v prvním sloupci.

Ve čtvrtém sloupci je uvedena délka podpisu (v bitech či bajtech) uváděna autory stažená na jednoho člena a při uvedeném klíči v šestém sloupci. Dále je zde uvedena doporučená (konkrétní počet) či odvozená velikost skupiny. Čili pro jaké velikosti skupin je schéma vhodné a efektivní.

V pátém sloupci je uvedena efektivita schématu, která určuje závislost délky podpisu a délky veřejného klíče v závislosti na velikosti skupiny, respektive počtu členů l. U některých schémat je uvedena i závislost stopování či odhalení identity na celkovém počtu členů ve skupině. Efektivita je nejčastěji uvedena jako lineární závislost na počtu členů, logaritmická závislost na počtu členů či konstantní závislost na počtu členů. Dále sou uvedeny informace o výpočetních operacích, nejčastěji se udávají pro protokoly podpis a ověření. Pokud autor přesné informace do návrhu neuvedl, pak informace v Tab. 1 schází či jsou odvozeny z jiného zdroje, který dané schéma analyzoval.

V šestém sloupci je uvedena bezpečnost schématu, použitý model a délka klíčů, modulů a jiných směrodatných hodnot. Nejčastější bezpečnostní model, který se u schémat vyskytuje je random oracle (model RO), který byl představen v [9]. Tento model tvoří mezikrok mezi teoretickou a praktickou kryptografií. U několika pokročilejších schémat se vyskytuje reálný model, tzv. standard model, kde jsou předpokládané omezené výpočetní a časové prostředky útočníka. Dále je předložena doporučená délka klíčů či jiných bezpečnostních parametrů, pokud jej autoři schémat uvedou.

Tab. 1: Přehled a základní vlastnosti schémat skupinových el. podpisů či jejich derivací.

Autoři, rok a odkaz v literatuře	**Typ**	**Kryptografická primitiva**	**Délka podpisu / Velikost skupiny**	**Efektivita podpisu / Výpočetní operace**	**Bezpečnostní model / Délka klíčů**
Ateniese, Camenisch, Hohenberger, Medeiros 2006 [1]	Kratké skupinové podpisy pro dynamické skupiny bez TTP	Asymetrické bilineární grupy, Silné LRSW, Silné SXDH a EDH	2052 b (2560 b) / Velké skupiny	Konstantní $O(1)$ / Otvírací algoritmus $O(n)$ a $O(\log n)$ při degeneraci efektivity podpisu na $O(\log n)$	Standard model / 170 b křivky z G_1 (256 b křivky)
Ateniese, Camenisch, Joye, Tsudik 2000 [2]	Skupinové podpisy odolné proti domluvě	Problém diskrétního logaritmu, SRSA, DDH, hašovací funkce	1087 B (8696 b) / Velké skupiny	Konstantní $O(1)$ / -	Model RO / RSA (klíč 1024 b)
Bellare, Micciancio, Warinschi 2003 [8]	Skupinové podpisy pro statické skupiny	NIZK a jednocestné funkce (existence trapdoor permutace)	- / Malé skupiny	Polynomická $O(\text{poly } n)$ / -	Standard model / -
Boneh, Boyen a Shacham 2004 [13]	Krátké skupinové podpisy se skupinovým a revokačním manažerem	Problém diskrétního logaritmu SDH s bilineárními mapami, Problém lineárního rozhodnutí, viz [13], hašovací fce a důkaz nulové znalosti	Menší než 200 B (1533 b) / Malé skupiny	Lineární $O(n)$ / Podpis: 8 umocňování Ověření: 6 multi-umocňování a jeden výpočet párování (náročnější než umocnění)	Model RO / 170 b křivky z G_1 a zanedbatelná pravděpodobnost padělaní
Boneh, Shacham 2004 [14]	Krátké skupinové podpisy s lokální revokací, zvýšená efektivita ověření se snížení anonymity	Problém diskrétního logaritmu SDH s bilineárními grupami, hašovací funkce	141 B (1128 b) / Malé skupiny	Lineární $O(n)$ / Podpis: 8 umocňování a 2 operace s bil.mapami Ověření: 6 umocňování a 3+2 operace s bil. mapami	Model RO / 170 b křivky z G_1 a zanedbatelná pravděpodobnost padělaní
Boyen, Waters 2007 [15]	Skupinové podpisy	Problém diskrétního logaritmu CDH, HSDH s bilineárními grupami, NIZK	- / Střední skupiny (cca stovky členů)	Konstantní podpis $O(1)$ Veřejný klíč roste logaritmicky na délce zpravy $O(\log n)$ / -	Standard model / -
Camenisch, Groth 2004 [23]	Skupinové podpisové schéma pro dynamické skupiny	Problém diskrétního logaritmu, problém faktorizace SRSA, hašovací funkce a důkaz nulové znalosti	- / vhodné pro střední skupiny (60 členů)	Lineární $O(n)$ / -	Model RO / RSA (klíč 2048 b) a hash SHA-1 (160b)
Camenisch, Lysyanskaya 2004 [29]	Skupinové podpisové schéma s důvěryhodnou třetí stranou	Problém diskrétního logaritmu DDH a LRSW s bilineárními mapami, kolizím odolná hašovací funkce	5296 b / Malé až střední skupiny	- / Podpis: 14 multi-umocňování a 3 násobení Ověření: 16 multi-umocňování a 3 operace párování s bil.mapami	Model RO / 171 b křivky z G_1 a zanedbatelná pravděpodobnost padělaní
Camenisch, Stadler 1997 [32]	Skupinové podpisové schéma	Problém diskrétního logaritmu, RSA, Schnorr, Hash fce	1,4 kB (11200 b) / Velké skupiny	Konstantní $O(1)$ / -	- / Hash 160 b Modulus 600 b

Furukawa, Imai 2006 [40]	Skupinové podpisové schéma	Problém diskrétního logaritmu DDH a SDH s bilineárními mapami, kolizím odolná hašovací funkce	1711 b / Malé až střední skupiny	- / Podpis: 4 multi-umocňování a 7 násobení Ověření: 5 multi-umocňování, 6 násobení a 2 operace párování s bil.mapami	Model RO / 171 b křivky z G_1 a zanedbatelná pravděpodobnost padělaní
Groth 2007 [42]	Skupinové podpisové schéma pro dynamické skupiny	Problém diskrétního logaritmu SDH a Problém lineárního rozhodnutí s bilineárními grupami, bez-kolizní hašovací funkce, NIZK	Menší než 2 kB / Velké skupiny	Podpis i klíče jsou konstantní $O(1)$ / -	Standard model / 256 b bilineární grupy (Veřejný klíč asi 1 kB) SHA-256
Chen, Pedersen 1995 [47]	Skupinové podpisové schéma	Problém diskrétního logaritmu, hašovací funkce a důkaz nulové znalosti	- / Malé skupiny	Lineární $O(n)$ / -	Teoretický koncept / -
Kiayias, Yung 2004 [50]	Skupinové podpisové schéma pro dynamické skupiny	Kombinace SRSA a DDH	- / Střední skupiny	Lineární $O(n)$ / -	Model RO / RSA (klíč 1024 b)
Li X., Qian, Li J. 2009 [55]	Demokratické skupinové podpisy s prahovým stopováním	Problém diskrétního logaritmu, hašovací funkce a důkaz nulové znalosti	- / Malé až střední skupiny	Podpis: lineární $O(n)$ Stopování lineární $O(t)$, kde $t < n$ / -	Teoretický koncept / -
Nguyen, Safavi-Naini 2004 [60]	Skupinové podpisové schéma	Problém diskrétního logaritmu SDH a DDH s eliptickými křivkami	597 B (4776 b) / Velké skupiny	Konstantní $O(1)$ / Podpis: 6 multi-umocňování a 20 násobení Ověření: 2 multi-umocňování a 3 operace párování a 13 násobení	Model RO / 170 b eliptické křivky
Wei 2005 [71]	Skupinové podpisové schéma se stopováním po dvojitém zneužití	SRSA, DDH, SDDH PedCom (3 návrhy)	- / Velké skupiny	Konstantní $O(1)$ / -	Model RO / -

2.2.5 Vývoj schémat skupinových podpisů

Skupinové podpisy byly uvedeny v CHVH91 [46], ve kterém se stanovily první vlastnosti, a navrhla se 4 schémata. Stanovené podmínky byly nedostatečné pro bezpečnost a schémata byla rovněž neefektivní, jelikož délka podpisu či délka skupinového veřejného klíče závisela na velikosti skupiny (počtu členů). Viz také CHP94 [47], které jen navíc přidává některé nové bezpečnostní vlastnosti, např. odolnost konstrukce.

Snaha o nezávislost délky podpisu, jeho zkrácení a zvýšení efektivity

Na počátku vývoje byla snaha o návrh schémat, které produkují délku podpisu a veřejného skupinového klíče nezávisle na velikosti skupiny. Schéma CS97 [32] již splňovalo dané kritéria. Schéma CM98 [31] bylo také vhodné pro velké skupiny. Víceméně však dávalo stále dlouhý

podpis (kolem 1000 B) a jeho registrační protokol byl málo efektivní. Schéma CD00 [22] například řeší, jak docílit úplné separability na skupinového manažera a revokačního manažera.

V následujících letech, po přelomu tisíciletí, se různě vylepšovala efektivita, zkoušely se různé kryptografické primitivity např. RSA, DL či eliptické křivky P02 [62]. Řešení efektivního lokálního ověření se objevilo v BS04 [14], které je bezpečné v random oracle (RO) modelu, viz [9]. Rovněž se pomocí některých nových kryptografických primitiv DDH, SRSA využívající v KY04 [50], či pomocí bilineárních map se zkoušelo dosáhnout efektivnější konstrukce v CG04 [23] a CL04 [29], bezpečné v RO modelu. Dále byla snaha o zkrácení délky podpisů, viz schéma BBS04 [13] založené na SDH a ZKPK.

V dalších letech se již autoři snažili přebrat to nejlepší z jednotlivých návrhů a snažili se dále zefektivnit konstrukci. Schéma BSZ05 [10] řešilo konstrukční problémy při přibírání či odebraní nového klienta. Schéma KY05 [51] se například zabývalo zlepšením bezpečnostního protokolu přidání nového klienta. Schéma FI06 [40] zefektivnilo konstrukci založenou na bilineárních mapách.

Přechod z RO modelu na standardní model

Do této doby byla většina navrhovaných schémat bezpečná v RO modelu (random oracle model, viz [9]). Avšak začaly se již v některých schématech objevovat i návrhy jak zaručit bezpečnost ve standardním modelu, který je podobný skutečnému prostředí, kde útočník má omezené časové a výpočetní prostředky. Jedno z prvních schémat, které prokazatelně zaručuje bezpečnost ve standardním modelu, je BMW03 [8], které je vhodné pouze pro statické skupiny. Ale mnohem efektivnějším schématem je ACHM06 [1], které bylo založeno na nových kryptografických předpokladech Strong SXDH (Symmetric External Diffie-Hellman), Strong LRSW (autoři Lysyanskaya, R.L. Rivest, A. Sahai and S. Wolf, Pseudonym Systems, SAC 1999) a EDH (Extended Diffie-Hellman), přičemž má nezávislou délku klíče na velikosti skupiny a je pouze o 35 % delší než BBS04 [13] s RO modelem.

Jiné schéma prokazatelně bezpečné bez RO BW06 [15] je postaveno na dvou úrovních. První úroveň pro bezpečný skupinový podpis založený na bilineárních grupách a druhá úroveň pro anonymitu využívá protokolu NIZK (Non-Interactive Zero Knowledge) ke skrytí identity. V BW06 [15] však velikost podpisu roste logaritmicky s velikostí skupiny. V dalším schématu BW07 [16] se již docílilo ke zkrácení velikosti podpisu na téměř konstantní velikost v závislosti na skupině, ačkoliv bylo toto schéma stále nevhodné pro skupiny s velkým počtem členů. I další schéma G07 [42] je bezpečné ve standardním modelu, bez RO. Toto schéma používá bilineární grupy, kde podpis je konstantní v závislosti na velikosti skupiny a umožňuje dynamické připojení nových členů.

Vylepšování algoritmů a vlastností schémat

Další schémata již dále upravovala jen některé části či algoritmy tak, aby zvýšila efektivitu. Například schéma LCHH09 [53] založené na předpokladech diskrétního logaritmu odstraňuje potřebu nové distribuce klíče členu, u kterého byla odkryta jeho identita, ale je žádoucí, aby jeho podpisový klíč platil nadále. Jiné schéma BCHLY08 [11] řeší čestné stopovatelné multi-skupinové podpisy, kdy uživatel může být zároveň členem více skupin. Navrhuje se zde, jak odradit uživatele skupin od šíření a nežádoucího sdílení tajných klíčů. Dále bylo navrženo odhalení identity jen při závažném důvodu, pomocí tzv. spravedlivých autorit. Některé nové schémata I09 [48], LQL09 [55] či SS09 [66], které se nazývají demokratické skupinové podpisy, řeší spravedlivější odkrývaní identity členů. To je docíleno prahovými schématy, kdy k odhalení identity jistého klienta je potřeba spojit určitý počet členů, tzv. podskupinu členů.

S postupem času přibývají nové vlastnosti jako například vyzrazení na svobodu (Leak-freedom) či okamžité odvolání (Immediate-revocation), které se objevily v DTX09 [36] a jsou důležité pro velké skupiny členů. Plné definice těchto dvou vlastností lze najít v [36].

Pro zefektivnění ověřovacího protokolu se začíná uvažovat o tzv. dávkovém ověřování, viz FGHP09 [38]. Metoda dávkového ověřování spočívá v ověření více podpisů najednou za pomocí několika technik. Lze několika násobně urychlit ověření podpisu jedné zprávy při ověřování celé série podpisů zpráv na rozdíl od jednotlivého ověření. Každá technika se liší v závislosti na konkrétním schématu. Můžeme například využívat shodu některých parametrů či vhodně pohybovat s exponentem, více v [38]. Při zvyšování počtu ověřovaných podpisů, tzv. dávky, se rychlost ověření na jeden podpis po nelineárním poklesu ustaluje.

S vývoje je patrné, že skupinové podpisové schémata jsou stále pokročilejší. Autoři se zabývají možnými vylepšeními jak celkových konstrukcí, tak i jednotlivých dílčích částí a algoritmů.

2.2.6 Zhodnocení skupinových podpisů

Skupinové podpisy byly dlouho vyvíjeny a stále jsou zlepšovány. Od počátečních teoretických návrhů, přes návrhy tzv. state of art, které jsou bezpečné v RO až po sofistikované návrhy, které přináší bezpečnost v reálném prostředí (standard model). Mezi nejpokročilejší návrhy lze zařadit BW07 [16] a schéma G07 [42], které je bezpečné ve standardním modelu, používá bilineární grupy a jeho podpis je konstantní v závislosti na velikosti skupiny, přičemž umožňuje dynamické připojení nových členů. Další schémata se snažila vylepšovat dílčí části a jednotlivé algoritmy. Například v FGHP09 [38] autoři navrhují zefektivnění ověřovacího protokolu pomocí tzv. dávkového ověřování. Nebo schéma LCHH09 [53] odstraňuje potřebu nové distribuce klíče členu, u kterého byla odkryta jeho identita neprávem. Navrhované koncepce se v podstatě snaží o tyto primární cíle: bezpečnost ve standardním modelu, efektivita hlavních algoritmů (zápis, podpis, ověření, trasování) a krátký podpis.

2.3 Další systémy s ochranou soukromí

Základní princip skupinových podpisů, kdy člen skupiny podepisuje data ve jménu skupiny, je vhodný u firem či různých organizací, kdy je zbytečné a leckdy nežádoucí ručit v el. podpisu osobními informacemi. Avšak vznikají systémy, kde rovněž je žádoucí zachovávat anonymitu např. anonymní přístupové a důvěryhodné systémy, elektronické mince a systémy el. hotovosti (v ang. E-cash) či anonymní elektronické petice (v ang. E-petition). Anonymní přístupové systémy budou představeny podrobněji dále v kapitole 3 a v kapitole 4, kde bude popsána jejich praktická implementace. Existují však i systémy, kde je anonymita nutností z hlediska společenských poměrů, např. elektronické hlasování (v ang. E-voting). Díky různým funkčním modifikacím se skupinové podpisy mohou aplikovat i ve výše zmíněných systémech.

2.3.1 Elektronické hlasování

Skupinové podpisy se dají vhodně aplikovat i pro systém elektronického hlasování (v ang. E-voting). Schémata skupinových podpisů musí zaručovat nezávislost délky el. podpisu na velikosti uživatelů, jelikož pracují s velkým počtem uživatelů (voličů). Dále je žádoucí, aby byl rozdělen skupinový manažer na dva samostatné orgány revokačního manažera a skupinového (zápisového) manažera. Takovéto podmínky nabízí schéma CS97 [32]. Dále je vhodné, aby anonymita byla odvolatelná pouze při dvojím hlasováním (analogie s dvojitým utracením

mince), schéma [71] má výše uvedené vhodné vlastnosti, které jsou uvedeny v Tab. 1. Dalším vhodným schématem je slepé podpisové schéma CKW04 [27], které má vlastnosti uvedeny v Tab. 2.

Princip el. hlasování je jednoduchý. Volič obdrží volební lístek (ballot) podobně jako by obdržel el. minci. Vybere si kandidáta a lístek podepíše. Skupinové podpisy zajišťují anonymitu. U el. hlasování je nutné zamezovat tzv. prodeji hlasovacího lístku útočníkovi, pomocí funkce aktualizace lístků, kdy útočník neví, zdali má v rukou platné či neplatné (monitorované) lístky. Podrobněji o elektronickém hlasování viz [44] či [56].

2.3.2 Kruhové podpisy

Uživatel si může libovolně zvolit jiného uživatele, kterého připojí do podpisového algoritmu. Každý uživatel má vlastní pár veřejný a privátní klíč. Podpis σ zprávy m vznikne podpisovým algoritmem se vstupem (m, SK_i, PK_1, ... , PK_n). Zpráva podepsaná kruhovým podpisem je tak dosvědčována uživateli ze skupiny. Validnost kruhového podpisu (v ang. ring signature) je dána σ, m, a veřejnými klíči uživatelů, PK_1, ... , PK_n. Ověřovatel je tak přesvědčen o validnosti podpisu, který vytvořil někdo ze skupiny, ale neví, o koho se jedná.

Od skupinových podpisů se kruhové podpisy liší ve dvou záležitostech. Neumožňují odhalit anonymitu uživatele, který inicializoval podpis a jakákoliv skupina uživatelů může být použita jako nová skupina bez dodatečného nastavení. Schéma SW07 [67] je navrženo pro standardní model a poskytuje podpisy o délce $2n+2$ lineárně závislé na délce skupiny n.

Systém TOR

Systém TOR, který je svým způsobem podobný kruhovému podpisu, dokáže zajistit částečnou anonymitu při použití určitých webových aplikací a schovávat IP adresu uživatelova uzlu, více viz [41]. Systém TOR prakticky zajišťuje anonymitu na síťové vrstvě v rámci Internetu.

2.3.3 Porovnání systémů

Skupinové podpisy jsou vhodné jako základní blok pro mnoho systémů, které zajišťují anonymitu. Jednotlivé schémata skupinových podpisů se podle svých vlastností pak zaměřují na různé systémy typu firemní podpisy, el. mince, el. hlasování, el. petice či anonymní pověřovací systémy. Každé použití má mírně odlišné požadavky. U firemních skupinových podpisů vystačíme se schématy, které nemusí být přímo pro velké skupiny členů. Naopak u el. peticí a el. hlasování potřebujeme koncepty a schémata pro velké skupiny. Vyplývá to z praktických vlastností, kdy počet zaměstnanců firem je mnohem menší než voličů či lidí podepisujících petice. Některé aplikace také požadují co nejkratší podpis a co nejrychlejší odezvu, kde není požadavkem velký počet uživatelů. Například schéma BBS04 [13] navrhuje krátký podpis pro systém VSC (Vehicle Safety Communications), viz [13], kde si mezi sebou blízké automobily zabezpečeně vyměňují výstražné upozornění.

V Tab. 2 jsou porovnávána schémata, která podobně jako schémata skupinových podpisů zaručují anonymitu a jsou také použitelné pro výše zmíněné systémy.

Tab. 2: Přehled a základní vlastnosti schémat systémů, které poskytují anonymitu.

Autoři a odkaz v literatuře	**Typ / Použití**	**Kryptografická primitiva**	**Délka podpisu / Velikost skupiny**	**Efektivita / Výpočetní operace**	**Bezpečnostní model / Délka klíčů**
Camenisch, Koprowski, Warinschi 2004 [27]	Slepé podpisové schéma / El. mince, el. hlasování	SRSA, kolizím odolná hašovací funkce a důkaz nulové znalosti	- / -	- / -	Standard model / RSA(klíč 1024 b)
Kiayias, Zhou 2008 [52]	Skryté el.podpisy založené na identitě / Firemní skupinové podpisy	Problém diskrétního logaritmu, SDH s eliptic.křivkami, DLDH hašovací fce	576 B / -	Podpis: 14 multi-umocňování a 2 operace párování Ověření: 10 multi-umocňování a 2 párování	170 b eliptické křivky, zvýšená efektivita ověření se snížení anonymity
Shacham, Waters 2007 [67]	Kruhové podpisové schéma	CDH a SDP	- / -	Lineární $O(2n+2)$ / Podpis: $2n+2$ Ověření: $2n+3$	Standard model / -

3 ANONYMNÍ AUTENTIZAČNÍ SYSTÉMY

Anonymní autentizační systémy AAS (v ang. Anonymous Authentication Systems) se také někdy označují jako anonymní pověřovací systémy (v ang. Anonymous Credentials Systems) či anonymní přístupové systémy (v ang. Anonymous Access Systems). Obecně fungují podobně jako skupinové podpisy, které byly představeny v kapitole 2. Jejich rozdíl je v použití, kdy u skupinových podpisů se podepisovala anonymně data jménem důvěryhodné skupiny, čímž se zaručila jejich validnost. Zatímco u anonymních autentizačních systémů určitá důvěryhodná autorita zaručuje validnost svého uživatele, tím že mu podepíše určité pověření (v ang. credentials) či token, který použije pro přístup k chráněným či placeným službám.

Tyto systémy v podstatě umožňují svým uživatelům anonymní přístup (v ang. access) k určitým zdrojům, aktivům, nebo také k chráněným místům. Uživatel či žadatel získá přístup k určitým aktivům na straně poskytovatele služeb (v ang. Service Provider SP) prostřednictvím podepsaného certifikátu (podepsané pověření či token) získaného od poskytovatele certifikátu. Poskytovatelem certifikátu většinou bývá důvěryhodná třetí strana (v ang. Trusted Third Party TTP), důvěryhodná veřejná autorita (v ang. Public Authority PA) či samotný poskytovatel služby. Certifikát se od běžného liší v nepoužití osobních informací o svém držiteli, ale jeho hodnoty při porušení jistých pravidel vedou k výpočtu a odhalení identity svého držitele. Jako problém zůstává protichůdnost dvou základních funkcí: úplné soukromí uživatele vs. vedení záznamů. Systémy jsou postaveny na anonymní či pseudonymní autentizaci, která může být založena na prokázání nulové znalosti, použití závazků (v ang. commitments) viz [35] či předložení skupinového podpisu, kterým uživatel zaručuje oprávněnost k určitým aktivům. Anonymní pověřovací systémy jsou popisovány a navrhovány v [3], [4], [5], [6], [7], [19], [29], [59], [65], [66], [69], [70] či [73].

3.1 Motivace, výhody a nevýhody

Důvodů proč chránit a utajovat identitu, osobní údaje či chování nebo návyky uživatele se najde celá řada. Anonymita je například na místě při komunikaci na krizových linkách nebo v chatujících místnostech společensky citlivých komunit (lidé nemocní rakovinou, A.I.D.S či jiné citlivé společenstva). Anonymita má své místo i ve zpravodajství v kontroverzních tématech nebo ve zdravotnictví při různém dárcovství. Dalším přínosem je utajení chování či návyků uživatele, například při používání např. tiskárny, pracovní stanice, vybavení nebo přístupu do chráněných prostor v různých institucích a firem, kdy prozrazení identity uživatele či zaměstnance není nutné k přístupu a někdy dokonce nežádoucí z pohledu uživatele. Systémy AAS jsou svým způsobem v použití podobné elektronickým mincím, které byly představeny v kapitole 2, kdy se podobně jako el. mince používá určité pověření (token), které zaručí uživateli přístup k chráněným službám či prostorům.

Hlavní výhodou je ochrana soukromí uživatele. Další výhoda, která plyne z konceptu identifikace nulové znalosti, který se většinou u systémů AAS používá, spočívá v bezpečnější autentizaci. Autentizace na bázi ZK má výhodu, že se při autentizaci nepřenáší samotné klíčové tajemství (např. heslo, nebo správná odpověď na konkrétní otázku) a tudíž se toto tajemství nedostává za normálních okolností mimo svého uživatele, čímž se zvyšuje bezpečnost autentizační metody. Avšak aby nedocházelo k neoprávněným úkonům ze stran anonymních uživatelů, musí být systém nastaven tak, aby zpětně odkrýval anonymitu a identifikoval nečestné a podvádějící uživatele. Nevýhodou je někdy pomalá zpětná vazba odhalení nekorektního chování nečestných uživatelů u některých systémů AAS, která tímto svádí k podvádění.

3.2 Schéma a princip

Obecný princip, který již byl naznačen na začátku této kapitoly, se u různých anonymních autentizačních systémů mírně odlišuje. Schémata AAS se mezi sebou liší v mnoha ohledech a to nejen dle použitých kryptografických primitiv.

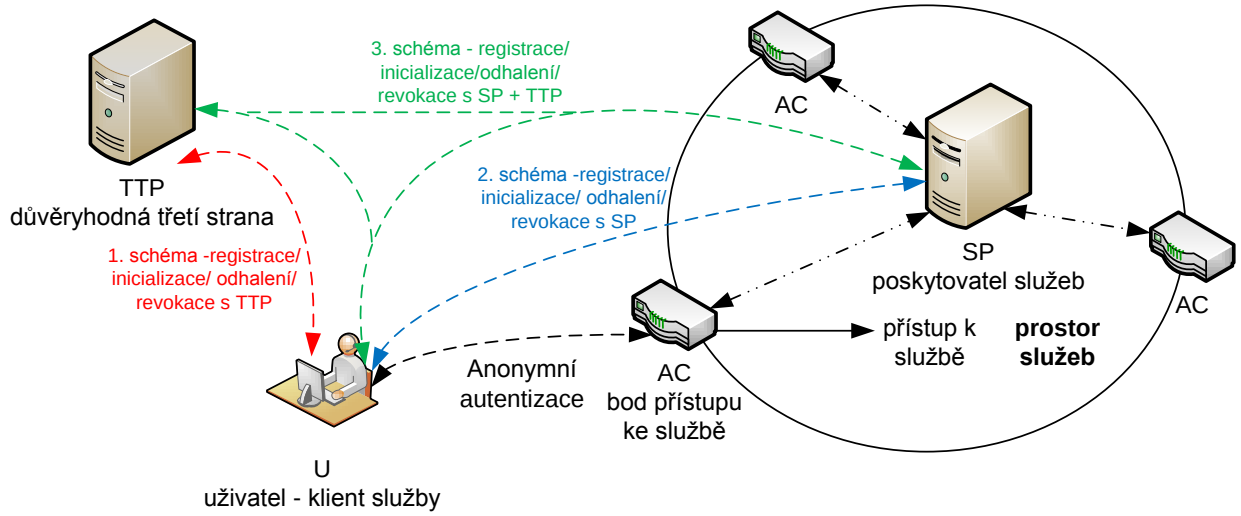
3.2.1 Druhy schémat AAS

Obecně lze rozdělit AAS na tři základní skupiny schémat. Schémata jsou ilustrována na Obr. 7, kde jednotlivé skupiny jsou barevně rozlišeny.

První skupina schémat AAS využívá důvěryhodnou třetí stranu, která ručí za validnost svých členů a zpětně může odkrýt jejich anonymitu. Obsahuje tak tři strany: uživatele U, důvěryhodnou třetí stranu TTP a poskytovatele služeb SP. Systém má nevýhodu, že veškerá moc a kontrola je na straně TTP, kde probíhá registrace uživatelů, inicializace pověření (el. podpis) a jejich revokace a odhalení identity.

Druhá skupina schémat AAS nepoužívá TTP a registraci anonymních uživatelů si zajišťuje samotný poskytovatel služeb SP, který pak sám odhaluje jejich identitu dle vlastních pravidel. Opět je zde nevýhoda jednostrannosti a potenciálního zneužití pravidel na straně nečestného poskytovatele služeb.

Třetí skupina schémat AAS používá stranu TTP, která kooperuje se serverem SP. Obvykle se provede registrace uživatele na straně SP, ale tokeny se inicializují na straně TTP, čímž pro případné odhalení identity uživatele musí dojít ke spolupráci obou stran SP + TTP. Tato skupina je z pohledu uživatele nejspravedlivější a vyžaduje k jeho odhalení spolupráci TTP a SP.



Obr. 7: Obecné schémata anonymních autentizačních systémů (komunikace barevně rozlišena).

3.2.2 Fáze AAS

Registrace uživatele

V této první fázi dochází k zaregistrování uživatele u TTP nebo SP. Uživatel musí věrohodně poskytnout vlastní identitu a osobní údaje. Bezpečnou možností je fyzická registrace, kdy se uživatel osobně prokazuje pomocí občanského průkazu, cestovním pasem nebo jiným důvěryhodným průkazem. Osobní údaje se uloží do databáze a spojí se s jedinečným identifikátorem (vysoké číslo), případně s číslem účtu.

Inicializace tokenu

V druhé fázi již nastupují kryptografická primitiva a vygenerují se veřejné parametry zvoleného kryptosystému (RSA, DL, eliptické křivky). Uživatel zde žádá o pověření (token) či několik tokenů, které získá bezplatně či platbou v rámci vlastního účtu od TTP či SP. V tomto bodě dochází k zahalení identity uživatele pomocí kryptografických protokolů jako konceptů nulové znalosti, slepých podpisů, závazků či jiných dle systému AAS. Uživatel tak získá pověření či token, který není přímo spojen s jeho identitou, či je jeho identita ukryta v tokenu a lze ji extrahovat jen prolomením nepoddajných kryptografických problémů. Aby SP či TTP spojili toto pověření s identitou, musí dojít k určitému porušení pravidel.

Autentizace

V této fázi se uživatel prokazuje tokenem nejčastěji u poskytovatele služeb, specificky na jeho přístupovém bodu (v ang. Access point), kdy žádá o jednu či více chráněných nebo placených služeb výměnou za token. Obvykle poskytovatel služeb zkontroluje platnost tokenu, tj. validnost jeho podpisu vydaného TTP či samotným SP, a dále se provede operace, která hraje zpětnovazební roli pro odhalení podvodníků např. při dvojitém použití či *k*-použití. Nejčastěji se používají kryptografické protokoly jako např. závazky nebo identifikace na bázi nulové znalosti (ZKPK, ZKIP nebo NIZK). Pokud vše korektně proběhne, uživatel získá přístup k službám.

Revokace

K revokaci tokenu může dojít přirozenou formou, kdy skončí datum jeho validnosti či se *k*-krát použije, nebo vynucenou formou při porušení určitých pravidel. Pokud nečestný uživatel poruší pravidla, budou jeho tokeny revokovány a zneplatněny pro fázi autentizace.

Odhalení ID

Tato fáze slouží k identifikaci podvodníků a uživatelů, kteří hrubě poruší jistá pravidla. Nejen že dochází k revokaci jejich tokenů, ale dochází také odhalení jejich identity a následného postihu.

3.3 Vývoj a typy anonymních autentizačních systémů

První systémy vycházely z myšlenky skupinových podpisů a elektronických mincí, kdy se pomyslná mince zobecnila na token či pověření, kterými se uživatel mohl autentizovat a získat tak přístup k chráněné službě. Novější anonymní autentizační schémata rovněž sledovala trendy skupinových podpisů či elektronických hotovostí a spoustu užitečných vlastností přebírala.

3.3.1 Vývoj AAS

Vývoj anonymních autentizačních systémů tedy procházel stejnými fázemi jako skupinové podpisy či systémy el. hotovosti, viz kapitola 2. Od prvních neefektivních návrhů CH91 [46], kdy velikost pověření či tokenu rostla v závislosti na velikosti skupiny uživatelů v systému, přes efektivnější, kdy velikost pověření již byla konstantní CL01 [28], CL04 [29], CV02 [33], až po současná schémata, kdy se zohledňuje bezpečnost kryptografických primitiv, efektivita výpočetních operací a snižovaní velikosti pověření pro použití na čipových kartách (smart card). Jelikož lze tokeny či pověření v nešifrované formě snadno zkopírovat a ukrást, například pomocí malware operujícího na uživatelské stanici, považuje se za bezpečnější ukládat tokeny na oddělená hardwarová úložiště tzv. TPM (Trusted Platform Module), viz BCC04[19].

3.3.2 Typy AAS

Obecně existuje mnoho typů anonymních autentizačních systémů. Některé typy AAS přebraly ze systémů elektronických hotovostí funkci elektronické peněženky a použily ji pro vícenásobné použití tokenů či pověření. Tímto se zefektivnila komunikace a snížily nároky na paměť pro ukládání tokenů. Systémy většinou využívají dynamický akumulátor, který redukuje dlouhou sadu hodnot *k*-tokenů na jednu hodnotu *v* pomocí jednosměrné funkce a po použití *k*-tokenů se musí hodnota *v* přepočítat, pro další validní operace. Mezi systémy, které dovolují použít pověření až *k*-krát, patří například ASM06[3], CHKLM06[24], NS05[59] či TFS04[69]. Občas se může stát díky špatné komunikaci, zpoždění, atd., že token je doručen například dvakrát k poskytovateli služeb. Proto třeba schéma CHKLM06[24] chrání anonymitu uživatelů, kteří neutrácí token příliš často, jde o tzv. glitch protection. Systém AKST08[4] přichází s podobnou vlastností, kde dochází k zablokování služeb od poskytovatele služeb po *d*-krát porušení pravidel bez odhalení identity.

Další typy AAS se zabývají otázkou vzájemné anonymity, kdy je v některých aplikacích vhodné chránit i identitu druhé strany (dalšího uživatele, autentizačního serveru či serveru poskytovatele služeb), jedná se o tzv. Peer-to-Peer AAS, viz SBM09[65], TS08[70] a ZSJ06[73]. Schéma SBM09[65] poskytuje mezi uživateli vzájemnou anonymitu, kteří díky poskytovateli služeb mohou uzavřít anonymní kontrakt tzv. Contractual Anonymity Protocol (CAP). Při porušení pravidel si poskytovatel služby vyžádá identitu od serveru s účty kontraktů.

Některá schémata, jako například BCKL07[6], se pokouší o praktický návrh neinteraktivního anonymního pověřovacího systému, jehož bezpečnost nezávisí na RO modelu. Jiná schémata, například SS09[66], přenáší vlastnosti prahových skupinových podpisů do systému AAS, kdy pro odhalení nečestných uživatelů je potřeba více stran či členů (zatím jsme uvažovali maximálně spolupráci SP + TTP). Další převzatou vlastností ze skupinových podpisů je delegace anonymního pověření, které bylo použito v systému BCCKLS09[7].

3.3.3 Implementace Idemix

Nyní je dostupná Open Source implementace anonymního pověřovacího systému Idemix, který je implementován v jazyku Java a popsán v [33]. Idemix umožňuje uživateli přesvědčit jakéhokoliv ověřovatele, že patří pod určitou společnost, aniž by odhalil jakéhokoliv další informace. Zejména ověřovatel pak nerozpozná pseudonym uživatele, pod kterým je znám u své vydavatelské společnosti. Analogický příklad z praxe: místo zobrazení uživatelova řidičského průkazu, který by odhalil jeho jméno, adresu, nebo věk, se odhalí pouze skutečnost, že on nebo ona má řidičský průkaz). V podstatě je systém Idemix rozšířením infrastruktury veřejných klíčů PKI o anonymitu. Každá organizace má veřejný klíč. Uživatel má odlišný veřejný klíč (pseudonym) pro každou organizaci. Uživatel vlastní podepsaný certifikát, který koresponduje s jeho veřejným klíčem. Při autentizace dokazuje jeho vlastnictví a vlastní certifikát neodhaluje. V konceptu kromě uživatele, organizace (vydávající certifikáty a ověřující jejich vlastnictví) existuje ještě organizace, která má pravomoc odhalit identitu uživatele. Kryptografické primitiva SRSA a DDH použité v systému Idemix vycházejí z [28].

3.4 Shrnutí

Anonymní autentizační systémy vycházející z teorie skupinových podpisů, nalézají široké uplatnění v anonymní komunikaci a v anonymních přístupových systémech k chráněným a placeným službám. Jako nejspravedlivější z pohledu uživatele se jeví schémata, kde je registrace, inicializace a odhalení identity rozděleno na dvě nebo více entit, nejčastěji na TTP

a SP, které navzájem spolupracují. Současné trendy se zabývají efektivním využitím *k*-násobného použití pověření nebo vzájemné anonymity obou stran. Byla také již navržena Open Source implementace anonymního pověřovacího systému Idemix, který je implementován v jazyce Java. Idemix je jeden z mála volně dostupných implementovaných návrhů, který zajišťuje anonymní autentizaci.

4 IMPLEMENTACE ANONYMNÍHO AUTENTIZAČNÍHO SYSTÉMU

Na fakultě FEKT VUT v Brně je vyvíjen systém Anonymous Authentication with Spread Revelation (**AASR**), který je představen v [43]. Systém využívá problému diskrétního logaritmu (PDL) a autentizaci na bázi nulové znalosti s čestným ověřovatelem (v ang. Honest Verifier Zero Knowledge HVZK). Praktickou částí této práce je implementace tohoto systému v programovém prostředí .NET a vytvoření aplikací v jazyce C#.

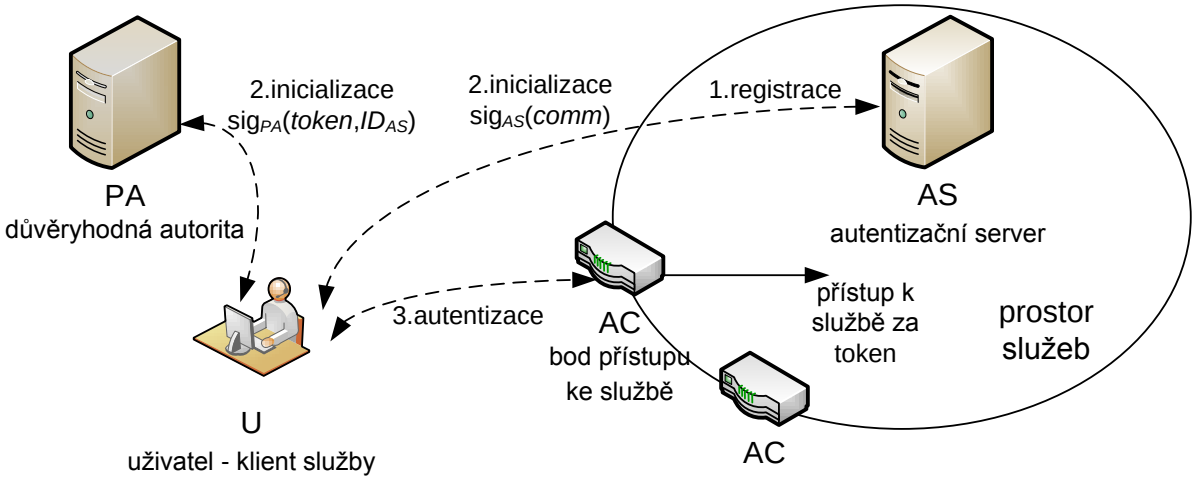
4.1 Systém AASR

Systém AASR je postaven na poskytování *tokenu* sloužícího k přístupu k digitální službě. *Token* je elektronicky podepsaná digitální informace a lze ho využít pouze jednou. Při jeho řádném použití nevyzrazuje žádná data vedoucí k identifikaci jeho majitele a v rámci jisté pravděpodobnosti ho nelze padělat. Systém AASR má rozdělenou moc pro odhalování identity uživatelů, revokaci a inicializaci *tokenů* mezi dvě entity: veřejnou autoritu (server PA) a autentizační server AS spadající pod poskytovatele služeb SP.

4.1.1 Schéma a konstrukce systému AASR

Na Obr. 8 jsou zobrazeny strany v systému AASR: uživatel (U), autentizační server (AS), ověřovatel-přístupový bod (AC) a veřejná důvěryhodná autorita (PA). Systém před udělením přístupu pracuje ve třech fázích. Nejprve proběhne **registrace**, kde se U fyzicky zaregistruje u AS a uloží se jeho *IDu* s veřejným klíčem. Poté probíhá **inicializace** (tolikrát, kolik si uživatel zaplatí tokenů), výsledkem U získá od AS podepsaný závazek *comm* obsahující tajný klíč *w*. Vytvoří si *token* a pošle ho PA, která kontroluje platnost podpisu od AS, konstrukci *tokenu* a vrací uživateli již použitelný podepsaný *token*. Pokud chce U využít chráněných služeb, musí proběhnout **autentizace** pomocí *tokenu*. Fáze autentizace probíhá prostřednictvím přenosných čipových karet (smart card).

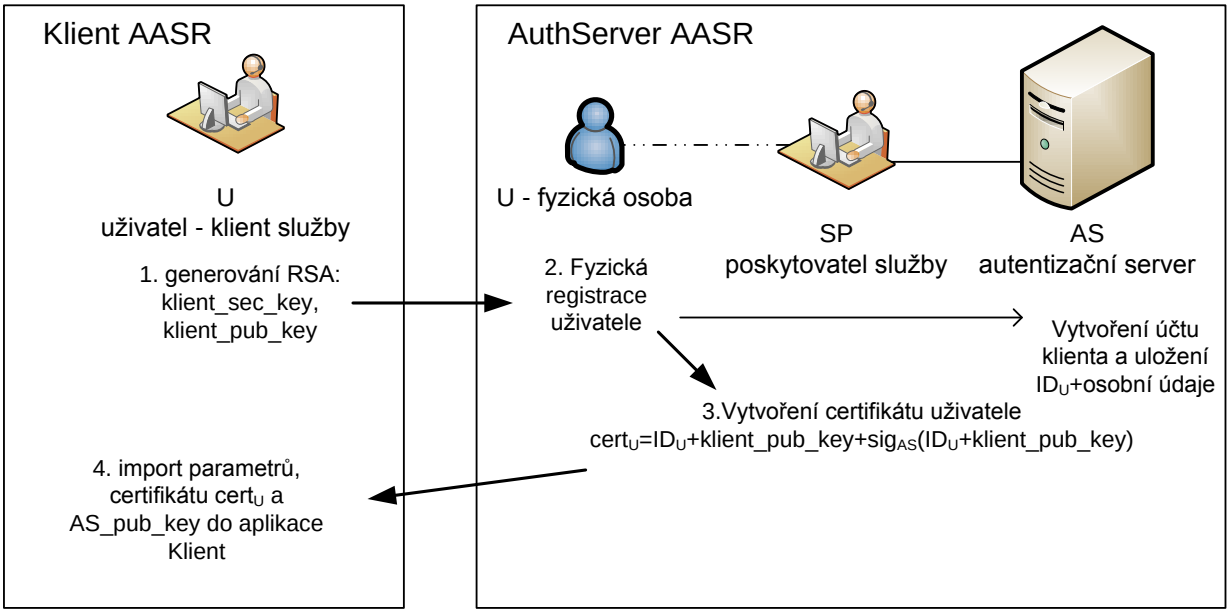
V této praktické části jsou vytvořeny aplikace na stranách U, AS a PA, které zajišťují vytvoření a generování bezpečných parametrů kryptosystému, fázi registrace, fázi inicializace, jejich vzájemnou komunikaci a předání podepsaných tokenů na čipovou kartu.



Obr. 8: Schéma systému AASR

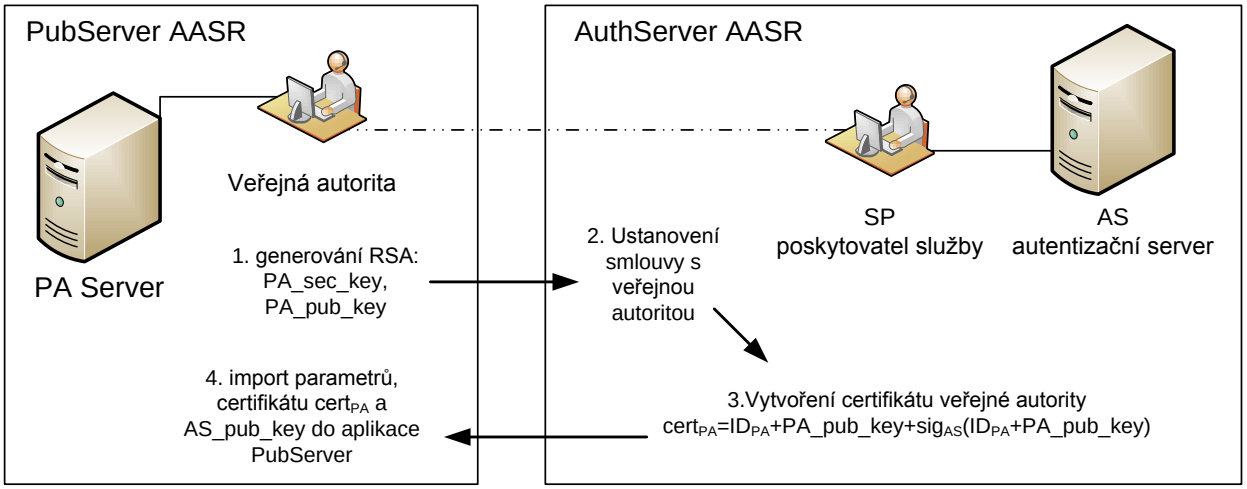
4.1.2 Registrace

Fáze registrace zahrnuje fyzickou registraci žadatele, který má zájem využívat anonymní systém AASR. Žadatel, který si nejprve vytvoří veřejný a tajný klíč RSA, přichází k poskytovateli služeb s vlastními osobními údaji (jméno, adresa, číslo občanského průkazu, veřejný klíč RSA atd.), kde je mu zřízen účet, který je zapsán do databáze na serveru AS a spojen s jedinečným identifikačním číslem ID_U. Od poskytovatele služeb získá AS_pub_key a certifikát $cert_U = ID_U$ + klient_pub_key + sigAS(ID_U + klient_pub_key) či certifikát X.509 a aplikaci Klient s již naimportovaným certifikátem a parametry systému AASR. Uživatel používá testovací certifikát $cert_U$ či X.509 při každé žádosti o *tokeny* v inicializační fázi. Celý postup je zobrazen na Obr. 9.



Obr. 9: Fáze registrace nového uživatele při použití vlastních certifikátů.

Před přidělování *tokenů* uživatelům v systému AASR musí dojít k jednorázové dohodě mezi poskytovatelem služeb SP a veřejnou autoritou PA, viz Obr. 10. Po uzavření smlouvy mezi SP a PA obdrží veřejná autorita certifikát od poskytovatele služeb $cert_{PA} = ID_{PA}$ + PA_pub_key + $sig_{AS}(ID_{PA}$+PA_pub_key), AS_pub_key a parametry systému AASR.



Obr. 10: Ustanovení dohody veřejné autority PA s poskytovatelem služeb SP.

Certifikace

V implementovaných aplikacích je možné vybrat ze dvou možností certifikace.

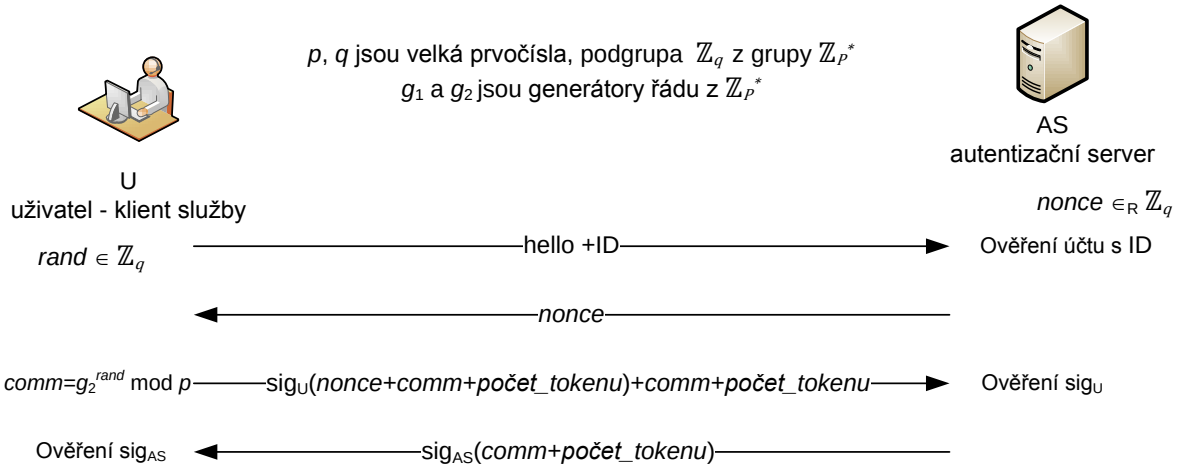
První možnost zahrnuje vlastní tvorbu certifikátů, které nejsou vztaženy k žádnému oficiálnímu standardu. Tyto certifikáty jsou v textové podobě v souborech **.txt*, které obsahují položky zobrazené na Obr. 9 a Obr. 10 (v bodech 3.). Certifikáty podepisuje samotná aplikace na straně serveru AS, která drží vlastně-podepsaný certifikát (v ang. self-signed) a k tomu tajný klíč RSA, kterým se certifikáty pro klienty a PA podepisují. Strana AS tedy tvoří vrchol řetězce důvěry a je zapotřebí střežit a bezpečně uchovávat tajný podepisující klíč RSA. Výhodou vlastní certifikace je přímá kontrola nad certifikáty. Nevýhodou je nestandardní formát certifikátů a nižší bezpečnost. Tato možnost je především určena pro testovací účely systému.

Druhou možností je využít certifikaci standardu X.509. Tato volba přináší větší bezpečnost a standardizovaný formát. Certifikáty pro testovací účely v systému AASR lze vytvářet pomocí utility *makecert.exe*, která je součástí Microsoft SDK. Klíče RSA se pak vytváří při tvorbě certifikátu a jsou uloženy v kontejnerech klíčů operačního systému Windows 7 či Windows Server 2008 R2. Certifikáty se ukládají do logických či fyzických úložišť, které se zvolí při generovaní certifikátu. Pro testovací účely je vytvořen kořenový certifikát AASR authority, kterým lze vytvářet klientské certifikáty a certifikáty serverů AS a PA, a který tvoří v řetězci důvěry certifikační autoritu.

4.1.2 Inicializace

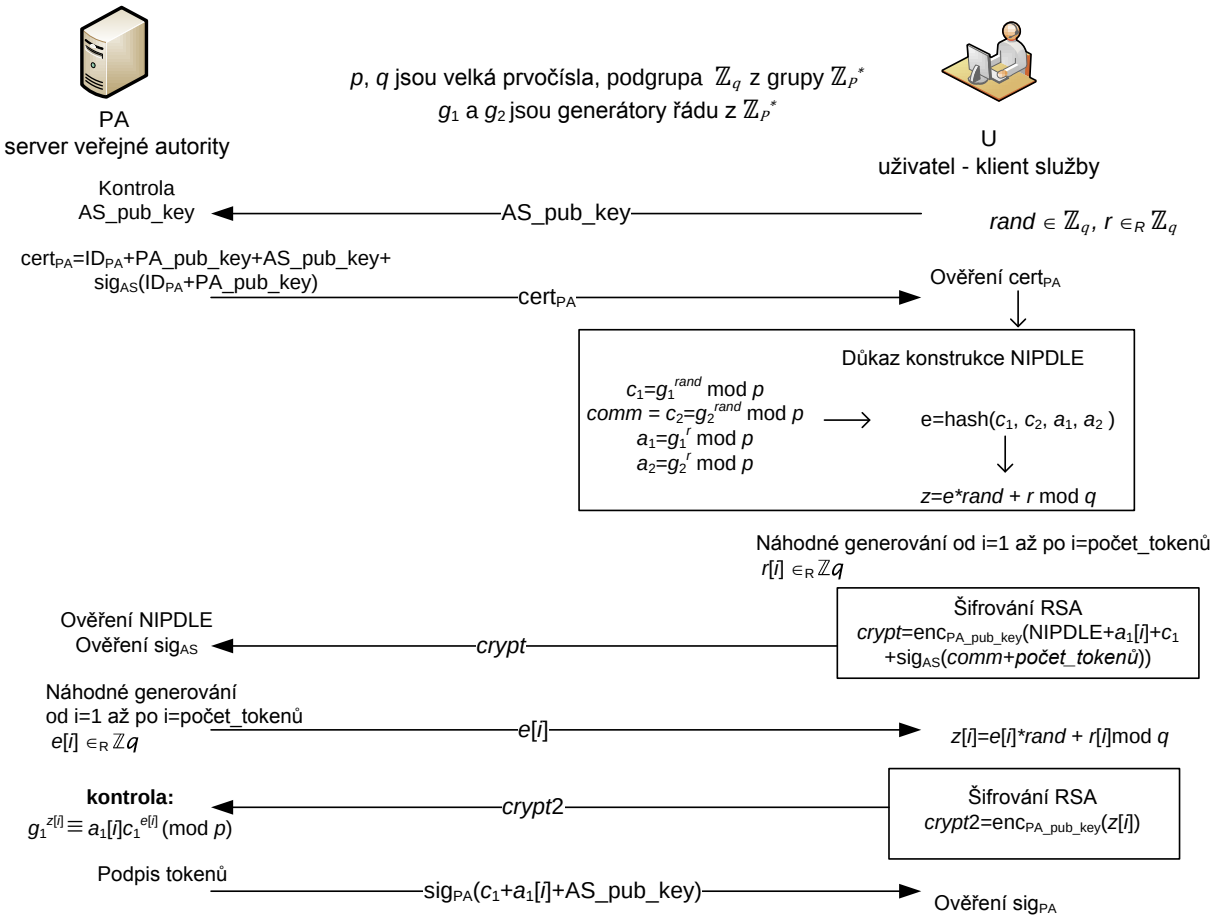
Fáze inicializace zajišťuje uživatelům přidělování (inicializování a registrování) tokenů bezplatně či platbou u poskytovatele služeb. Je rozdělena na dvě části. Implementovaná fáze inicializace se z testovacích důvodů v programech Klient, AuthServer a PubServer mírně liší. Například zde dochází také k přenosu parametrů p, q, g_1 a g_2 pro rychlejší testování.

V první části probíhá komunikace mezi klientem U a autentizačním serverem AS. Klient posílá hodnotu ID. Authentizační server najde podle hodnoty ID certifikát a účet klienta. Poté posílá klientovi náhodné číslo *nonce*. Klient pomocí importovaných parametrů a vlastního náhodně vygenerovaného tajného identifikátoru *rand* vypočítá závazek *comm*. Tento závazek spolu s obdrženým *nonce* od AS a číslem udávající *počet tokenů*, tj. o kolik jich má klient zájem, podepíše vlastním tajným klíčem RSA a zašle tyto hodnoty spolu s jejich podpisem na AS. Autentizační server nyní ověří podpis, a pokud je vše v pořádku, zaznamená do databáze *comm, počet tokenů* a případně proběhne platba (odečet z účtu vedeným pod ID_U). Nakonec AS posílá zpět uživateli zprávu *comm* + *počet tokenů* tentokrát podepsanou vlastním tajným klíčem (AS_sec_key). Postup je znázorněn na Obr. 11.



Obr. 11: První část inicializace, komunikace klient-AS.

Druhá část zahrnuje komunikaci mezi veřejnou autoritou PA a klientem U. V této části klient nejprve posílá veřejný klíč AS. Na základě tohoto klíče bude server PA předpokládat komunikaci v rámci systému AASR a pošle svůj certifikát klientovi, který si tak ověří platnost certifikátu serveru PA. Uživatel nyní vygeneruje náhodné hodnoty $r[i]$, pro každý token jedno r a sestaví tokeny $a_1[i]$, c_1. Dále provede neinteraktivní důkaz konstrukce NIPDLE (v ang. Non-Interactive Proof of Discrete Logarithm Equivalency) tak, že z hodnot (c_1, c_2, a_1, a_2) vypočítá haš e a na ni sestrojí odpověď z. Uživatel si z certifikátu vezme veřejný šifrovací klíč RSA PA serveru
a zašifruje důkaz NIPDLE, tokeny a podpis od AS (*comm* + *počet tokenů*) a kryptogram posílá veřejné autoritě PA. Server PA nyní rozšifruje a ověří důkaz NIPDLE, tím že sám provede nový výpočet haše e a ověří ho. Pokud se zprávy rovnají, ověří ještě pravost podpisu od AS nad závazkem *comm* a *počtem tokenů*. Při úspěšném splnění všech kontrol vygeneruje nyní pro každý *token* jednu náhodnou hodnotu $e[i]$ a tyto hodnoty odesílá uživateli. Uživatel nyní vypočte zprávy $z[i]$ pro každou výzvu $e[i]$, díky příslušným tajemství *rand* a $r[i]$, a odesílá zprávy zpět k veřejné autoritě. Ta nyní zkontroluje korespondenci s hodnotami $a_1[i]$, c_1 a validní tokeny spolu s veřejným klíčem AS (sloužící jako identifikátor systému AASR při autentizaci tokenů) podepisuje vlastním tajným klíčem RSA. Podpisy vrací uživateli, který zkontroluje jejich validnost a hodnoty si uloží do zašifrované databáze či raději bezpečně vyexportuje hodnoty na čipovou kartu. Matematický popis a postup druhé části inicializace je zobrazen na Obr. 12.



Obr. 12: Druhá část inicializace, komunikace klient-PA.

4.1.3 Autentizace, revokace a odhalení

V této sekci jsou krátce popsány fáze autentizace, revokace a odhalení, které však implementované aplikace neobsahují.

Autentizace

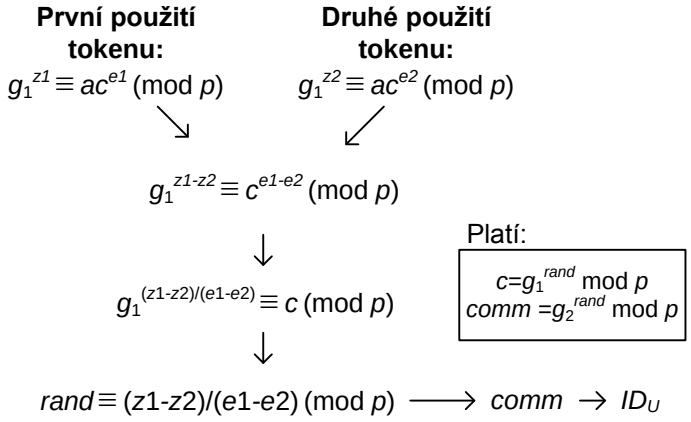
Tato fáze využívá ∑-protokol, konkrétně Schnorrův protokol pro autentizaci ZKPK, více v [64]. Uživatel prokazuje znalost tajemství *rand* tím, že správně odpoví pomocí *z* na výzvu *e* a AC ukládá hodnoty pro AS. Matematický popis se shoduje s Obr. 12 od úseku Ověření NIPDLE. Pokud nečestný U zneužije *token*, AS spolu s PA odhalí *rand* vedoucí k ID_U a k jeho postihu. Zde však předpokládáme zcela čestného ověřovatele AC, kterému klient musí důvěřovat. Výhodou je vysoká efektivita (použitelné na čipové karty). Fázi autentizace pomocí čipové karty řeší práce viz [49].

Revokace

Revokace tokenů neboli zrušení jejich platnosti, může být zajištěna přidáním časových omezení do podpisů AS a následně PA. Ovšem uživatel může použít jeden token pouze jednou, pokud chce zůstat v anonymitě. Pokud toto pravidlo poruší, lze odhalit jeho identitu a odvolat zbytek tokenů, kdy se pro tento krok musí spojit strana AS se stranou PA. Revokace může nastat i při porušení jiných pravidel, které by se ujednaly ve smlouvě mezi klientem a AS, a ve smlouvě mezi AS a PA.

Odhalení identity

Nečestný uživatel může být snadno odhalen. Například při druhém použití *tokenu*, server AS může snadno z uchovaných hodnot c_1, a_1, a_1', e_1, e_2, z_1 a z_2 snadno vypočítat uživatelovo *rand* a vypočítat závazek *comm*, který je spojen s jeho identitou v databázi. Pokud uživatel použije token pouze jednou (tj. server AS má pouze c_1, a_1, e_1, z_1), ale dojde k porušení jistých ujednaných pravidel, může server AS, respektive poskytovatel služby zahájit spolupráci s veřejnou autoritou (AS zašle na PA c_1, a_1), která po zvážení oprávněnosti žádosti vrací serveru AS hodnoty e_2 a z_2, díky kterým již AS může vypočítat *comm* a zjistit identitu nečestného či škodlivého uživatele. Postup výpočtu je znázorněn na Obr. 13.



Obr. 13: Matematický postup získání identity nečestného uživatele.

4.2 Autentizační server

V systému AASR při praktickém nasazení je autentizační server AS ovládán poskytovatelem služeb, respektive vydavatelem tokenů. Poskytovatel pak generuje veřejné parametry kryptosystému, registruje klienty, přidává, upravuje či odebírá jejich záznamy v databázi a vydává a podepisuje certifikáty a tokeny. Pro tyto a další účely v rámci praktické části diplomové práce je naimplementovaná aplikace AuthServer v prostředí .NET v jazyce C#. Statická třída Program implicitně spouští třídu AuthServerForm, která tvoří grafické rozhraní a obsluhuje funkce aplikace. V případě potřeby spouští třídy AuthServer, Crypto a druhé grafické rozhraní, třídu Databaze_certifikace, pro práci s databází a certifikáty.
Náhled na grafické rozhraní hlavního okna aplikace zobrazuje Obr. 14. Po spuštění **Registrování a databáze** se zobrazí okno, viz Obr. 15, které nabízí databázové funkce, registraci klienta a podpis vlastních certifikátů.

Aplikace AuthServer poskytuje následující hlavní funkce:

- generace veřejných parametrů (prvočíselný modulus *p*, prvočíselný modulus *q*, generátory g_1 a g_2), ukládání generovaných parametrů a zpětné načítání ze souboru *parametry.txt*,
- generovaní nových klíčů RSA a jejich export do kontejneru OS nebo do *klice.txt*,
- autentizace uživatelů (správců) loginem a heslem - chránění klíčových funkcí aplikace přihlašováním a odhlašováním, hašovací funkce pro heslo je SHA512,
- podepisování klientských žádostí o tokeny (*comm* + *počet tokenů*)
- podpis a tvorba vlastních certifikátu **.txt*,
- registrování nových klientů a jejich přidávání do databáze ODBC.ACCESS,
- editace údajů o klientovi v databázi ODBC.ACCESS,
- odebraní klienta s databáze ODBC.ACCESS,
- evidování počtu registrovaných tokenů každého registrovaného uživatele v databázi OBDC.ACCESS,
- možnost tvorby textové databáze obsahující údaje o klientech.

4.2.1 Generování parametrů

Před přijímáním žádostí o tokeny se nejprve musí vygenerovat parametry (tl. **Vygenerovat parametry**) systému AASR pomocí třídy Crypto. Zvolený řád ovládán posuvným ovladačem uvádí pouze orientační údaj a vygenerovaný řád parametrů se může mírně odlišovat díky náhodnému generátoru. Pro testovací účely je možné zvolit i generování malých parametrů. Malé parametry (tl. **Malé par.**) jsou prvočísla *p, q* v řádech 2^{10} až 2^{20}.

Pokud použijeme symetrické parametry, pak střední parametry *p, q* (tl. **Střední par.**) se pohybují mezi 2^{100} až 2^{200}. Velké parametry *p, q* (tl. **Velké par.**) jsou řádu 2^{1000} až 2^{1200}. Vysoké prvočísla velkých parametrů již zajišťují potřebnou bezpečnost, která je založena na problému diskrétního logaritmu. Při implicitně nastavených asymetrických parametrech *p, q* nám pro bezpečné hodnoty postačuje generování středních parametrů, kde se generuje *q* ve zvolených řádech 2^{100} až 2^{200} a řád prvočísla *p* je posunut až do hodnot 2^{1000} až 2^{1200}. Doporučené mezní hodnoty jsou $q > 2^{160}$ a $p > 2^{1024}$. Tyto asymetrické hodnoty pak mají výhodu nízkých hodnot *q*, *rand* a *r*, se kterými je počítána hodnota *z* ve fázi autentizace na čipové kartě, která kvůli svému omezenému výpočetnímu výkonu nižší hodnoty uvítá. Rovněž lze asymetricky vygenerovat i vysoké parametry, kde *q* je v řádech 2^{1000} až 2^{1200} a *p* v řádech 2^{2000} až 2^{2200}. Tyto hodnoty jsou více bezpečné, ale prodlužuje se doba registrace a inicializace tokenu.

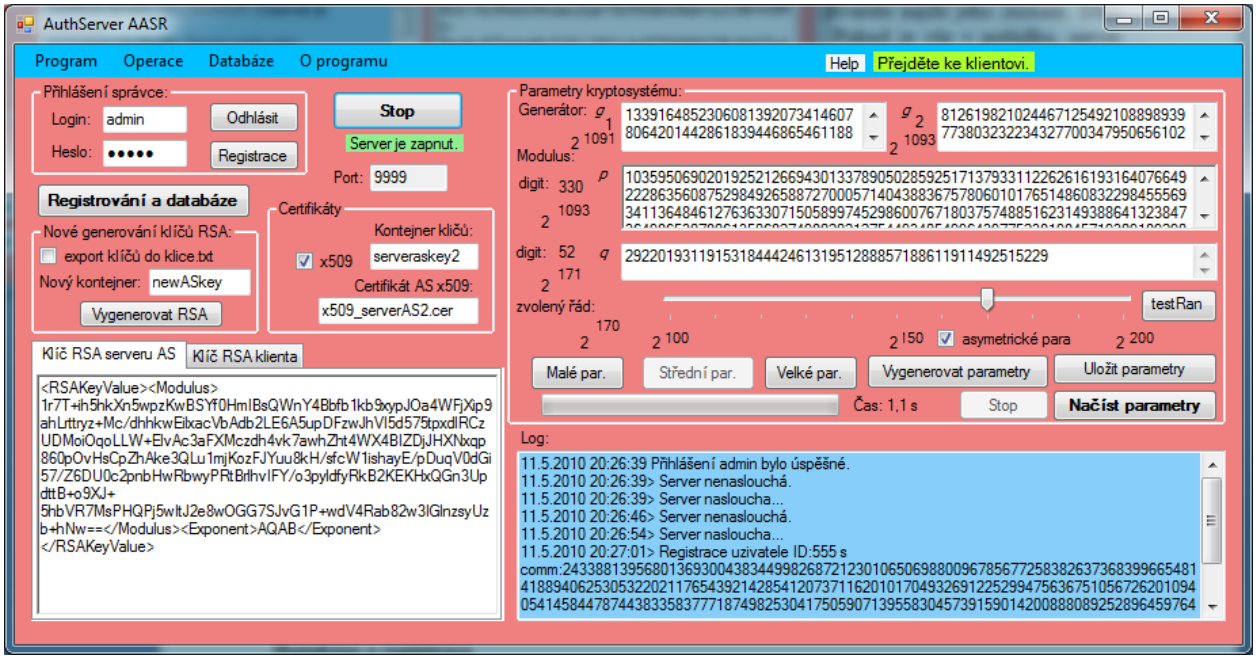
Generování malých a středních parametrů je zpravidla rychlé (do 1s). Čas generování však závisí na vhodnosti náhodných čísel, které musí být prvočísla *p*, *q*. Prvočíselnost je kontrolována Rabin-Millerovým testem prvočíselnosti, který je již součástí tzv. Open Source knihovny *emil.GMP.dll*, která je dostupná z [74].

Dalším aspektem, který parametry musí splnit, je test generátorů g_1 a g_2. U symetrických parametrů jsou iteračně hledány, dokud není splněna podmínka $g_i^q \bmod p=1$. U velkých symetrických parametrů se vyskytuje časový problém hledání generátorů, pokud jsou prvočísla řádově příliš odlišná. Pokud se doba generování stává neúnosnou je k dispozici funkce stop (tl. **stop**), které zruší generování a poskytne možnost pro nové generování, při kterém se vrací jiné prvočísla, které mohou rychleji poskytnout generátory. Navíc funkce stop dává příležitost ke změně řádu parametrů.

Pokud bychom však stejným způsobem hledali střední či vysoké asymetrické parametry *p* a *q*, narazili bychom na časově nesplnitelný úkol při iteračním testování podmínky $g_i^q \bmod p=1$. Proto je podmínka hledání generátorů pozměněna na stanovení $g_i = h^{(p-1/q)} \bmod p$, kde *h* je libovolné číslo splňující $1 < h < p - 1$. Nevýhodou tohoto určení generátorů je, že se nám zvýší jejich hodnota (přiblíží se řádu *p*). Výhodou je však okamžité nalezení generátorů. Po vygenerování parametrů již můžeme zapnout naslouchání serveru (tl. **Spustit server**).

4.2.2 Server naslouchá

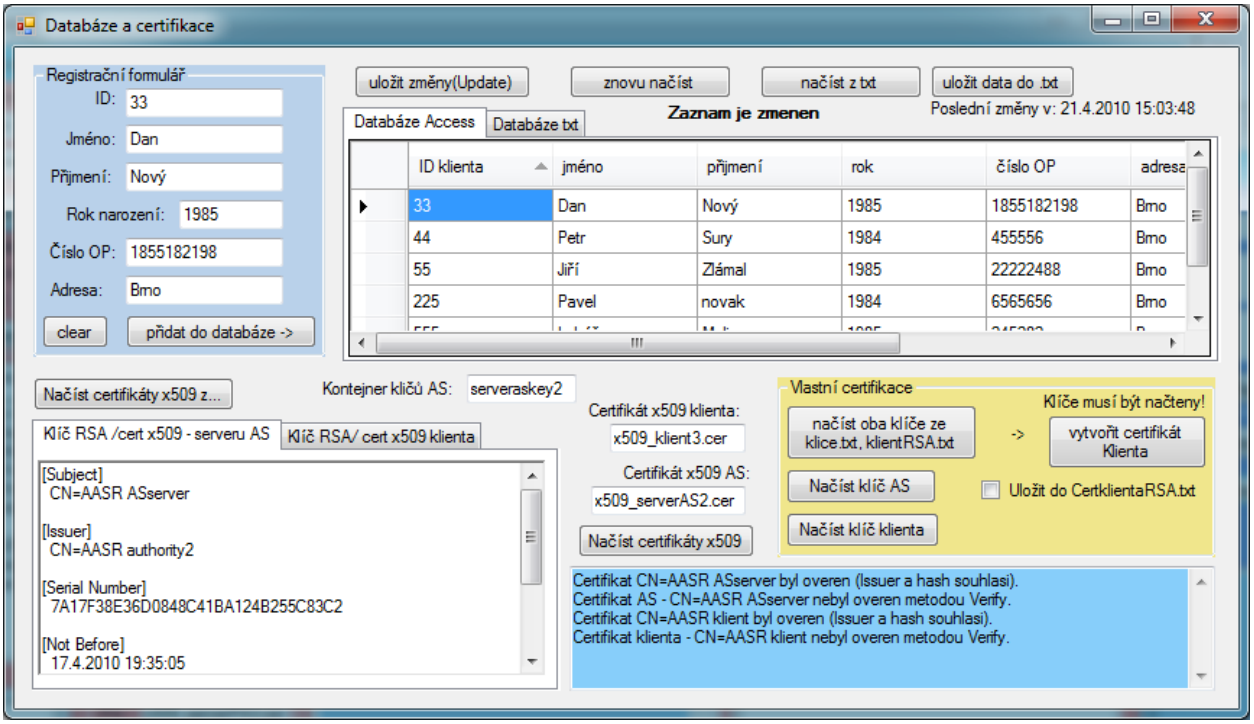
Pokud je server AS spuštěn a naslouchá, očekává příchozí provoz ze strany klienta. Pokud klient vyšle požadavek na tokeny, přijme server první jeho autentizační certifikát. Dle druhu první zprávy *helloVlastní* či *hellox509* rozezná použitou certifikaci. Díky certifikátu autentizuje uživatele a prohledá databázi ODBC.ACCESS, kde dle *ID* uživatele najde jeho záznam. Dále proběhne inicializační fáze, která je již popsaná v části 6.1. Pokud je vše v pořádku, server eviduje počet tokenů v klientově účtu v databázi a uloží si jeho závazek *comm*. Po odeslání podepsaného závazku *comm+počet tokenů* naslouchá dalšímu provozu.



Obr. 14: Náhled na grafické rozhraní hlavního okna aplikace AuthServer.

4.2.3 Databáze a registrace klienta

Na Obr. 15. je znázorněno databázové a registrační okno aplikace. Nového uživatele lze do databáze vložit přes registrační formulář po vyplnění všech položek a načtení jeho klíče. Dále je zde možnost vytvořit uživateli vlastní certifikát, který mu vystavuje server AS tvořící také soukromou certifikační autoritu. Také je možné přímo v zobrazení databáze měnit a mazat data. Databázi můžeme také externě měnit pomocí MS ACCESS v souboru *klienti*.



Obr. 15: Náhled na grafické rozhraní aplikace AuthServer, okno - Databáze a certifikace.

4.3 Klient

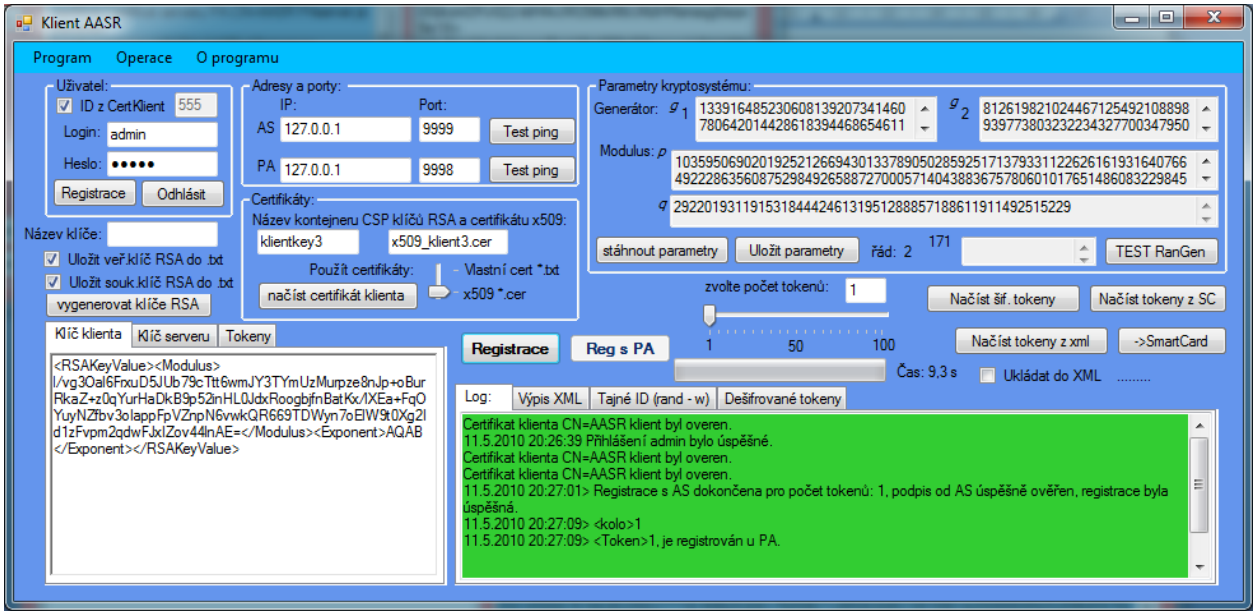
Aplikace na straně uživatele ProjektAASRv1 (zděděné jméno po Solution v .NET, ale pro přehlednost budeme dále označovat aplikaci jako Klient, kvůli svému využití) má za úkol registrovat a inicializovat tokeny u serverů AS a PA. Dále validní tokeny vhodně předává čipové kartě nebo exportuje tokeny ve formátu XML.

Aplikace Klient poskytuje následující hlavní funkce:

- autentizace uživatelů loginem a heslem - chránění klíčových funkcí aplikace přihlašováním a odhlašováním, hašovací funkce pro heslo je SHA512,
- generovaní nových klíčů RSA a jejich export do kontejneru OS nebo do *klice.txt*,
- testovaní dostupnosti serverů AS a PA funkcemi ping,
- načítání vlastních certifikátů nebo certifikátů X.509,
- inicializace tokenů u AS a registrace tokenů u PA serveru,
- opakovaná registrace u PA serveru při nezdaru komunikace s PA,
- implicitně ukládá zašifrované tokeny (RSA 2048 bitů) a zpětně je dešifruje po načtení,
- po povolení ukládá tokeny do XML (nešifrovaně) a zpětně je načítá,

- exportuje tokeny na čipovou kartu a vypisuje obsah čipové karty,
- ukládá a načítá použité parametry p, q, g_1 a g_2,
- logování a informování o průběhu komunikace.

Na Obr. 16 je zobrazeno grafické rozhraní aplikace Klient. Aplikaci po spuštění obsluhuje statická třída Program, která spouští implicitně grafické rozhraní KlientForm, kterým se ovládá celá aplikace a spouští další třídy Klient a Crypto. Třída Klient řeší komunikaci se servery AS a PA a inicializační fázi (inicializace a registrace tokenů). Třída Crypto implementuje metody RSA a generátory náhodných čísel. Veřejné struktury obsahují pouze veřejné parametry. Privátní a tajné parametry (například *r* či *rand*) jsou pro vyšší bezpečnost šifrovány RSA (2048 bitů) a ukládány do souboru. Pro získání těchto tajných hodnot musíme tyto hodnoty opět dešifrovat soukromým klíčem RSA, který je uložen v kontejneru klíčů OS. I aplikace Klient používá pro výpočty vysokých hodnot Open Source knihovnu *emil.GMP.dll*, která je dostupná z [74].



Obr. 16: Náhled na grafické rozhraní aplikace Klient.

Vlastní certifikace a X509 certifikace

Z praktických a bezpečnostních důvodů je povolen jen jeden režim certifikace, který lze současně při provozu využívat. Pokud bychom tedy chtěli použít neimplicitní vlastní certifikaci, musíme vypnout certifikaci X.509, tj. přepnout na vlastní a importovat klíče (tl. **načíst certifikát klienta**). Jelikož vlastní certifikace slouží pro testovací účely, má uživatel přístup i k serverům, u kterých musí rovněž naimportovat klíče pro vlastní certifikaci. Správnost nastavení indikuje shodný veřejný klíč RSA serveru AS na všech stranách systému AASR.

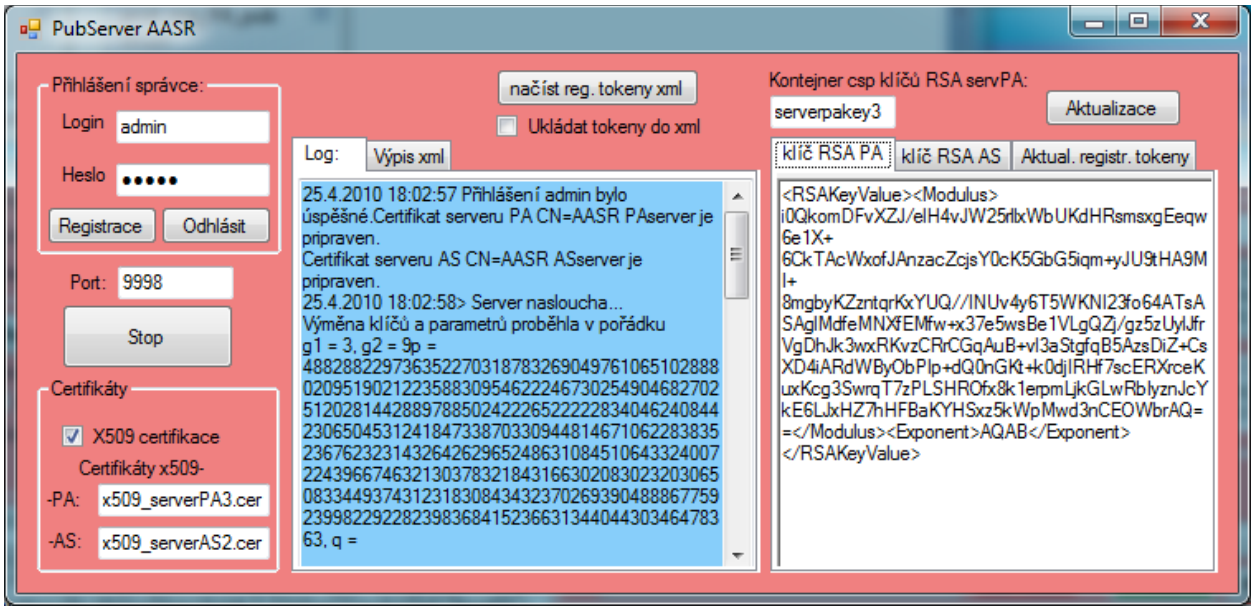
4.4 Server veřejné autority

Server veřejné autority PA disponuje aplikací PubServer. Jeho úkolem je především kontrola podpisů tokenů od AS, kontrola jejich konstrukce NIPDLE, kontrola první útraty každého tokenu a jejich podpis soukromým klíčem PA.

Aplikaci PubServer obsluhuje statická třída Program, která spouští implicitně třídu grafického rozhraní PubServerForm, kterou je ovládána aplikace. Třída PAserver řeší komunikaci s klientem inicializační fázi (tj. registraci tokenů). Třída Crypto implementuje metody RSA a generátory náhodných čísel. Rovněž i aplikace PubServer používá pro výpočty vysokých hodnot knihovnu *emil.GMP.dll*, která je dostupná z [74]. Na Obr. 17 je zobrazeno grafické rozhraní aplikace PubServer.

Aplikace PubServer poskytuje následující hlavní funkce:

- autentizace uživatelů (správců) loginem a heslem - chránění klíčových funkcí aplikace přihlašováním a odhlašováním, hašovací funkce pro heslo je SHA512,
- načítání vlastních certifikátů nebo certifikátů x509,
- načítání registrovaných tokenů v XML,
- registrace tokenů (jejich kontrola a podpis),
- ukládání registrovaných tokenů do XML souboru,
- logování a informování o průběhu komunikace.



Obr. 17: Náhled na grafické rozhraní aplikace PubServer.

Při nezaškrtnutém políčku **Ukládat tokeny do xml** server PA neukládá informace o registrovaných tokenech. Pro zpětné zjištění identity nečestných klientů AASR je nutné mít toto políčko vždy zaškrtnuté. Nevýhodou je, že databáze registrovaných tokenů v podobě XML narůstá lineárně s počtem registrovaných tokenů a může se stát, že překročí hranice paměti fyzického úložiště (pevného disku). Řešením je exportovat data XML na jiné úložiště a vymazat všechny dětské uzly (v ang. child nodes) uvnitř XML a nechat tokeny znovu ukládat do XML. Je nutné mít na paměti, že metoda pro ukládání tokenů potřebuje soubor *tokeny.xml*, který již obsahuje kořenový uzel.

4.5 Vstupy, výstupy a bezpečnost implementace

Tato část popisuje datové vstupy, výstupy a bezpečnost implementovaného systému AASR, respektive jeho aplikací Klient, PubServer a AuthServer.

Vstupy

Mezi vstupy implementovaného systému AASR lze zařadit všechny načítané proměnné ze vstupních formulářů (tj. *textbox, maskedbox* a *richbox*), dále načítané parametry a data ze souborů *.txt, *.xml, *.mdb či různých formátů certifikátů *.cer, atd. Tyto vstupní data a parametry jsou tvořeny přímo uživatelem. Dle příslušných dat musí uživatel dodržovat formáty a typy vstupních parametrů. Nesprávné hodnoty, tam kde je to vhodné, jsou ošetřeny chybovými hláškami a výzvami k napravení či změnění vstupních dat.

Dalším speciálním a důležitým vstupem systému je otevřené TCP spojení na příslušných portech na jednotlivých serverech AS a PA. Porty jsou implicitně nastaveny na hodnotě 9998 u PA serveru a na 9999 na AS serveru. Při změně portů se musí uživatel vyvarovat hodnot 0 - 1024, které používají obvyklé síťové služby. Při testování systému na LocalHostu nebyly funkční ani porty 10000 a 10001, které si zabírají různé webové relace atd. Doporučeno je využít tzv. *dynamické* a *soukromé porty* – v rozsahu 49152 až 65535.

Při samotné komunikaci mezi aplikacemi Klient - AuthServer a Klient - PubServer, před předáváním parametrů atd. probíhá autentizace certifikáty. Nestandardní situace a chyby, které se vyskytují během komunikace, jsou logovány oběma stranám.

Výstupy

Výstupy systému mohou tvořit stejně jako vstupy i formuláře a výpisové textové boxy. Dále jsou zde úložné soubory *.xml, *.txt, *.mdb, které mají za úkol uchovat data a databázi. Utajované data (samotné tokeny) jsou šifrovány RSA (2048 b). Při práci v důvěryhodném prostředí a v testovacích režimech pro urychlení náhledu na parametry inicializovaných a registrovaných tokenů, lze zvolit i nešifrované ukládání do *tokeny.xml* na straně klienta. Jedná se však o bezpečnostní riziko. Stejně tak by uživatel neměl exportovat při generování RSA oba klíče do souboru *klice.txt*, ale raději implicitně využít klíčového kontejneru, který poskytuje samotný OS (testováno na Windows 7 Professional). Nicméně je možnost vyexportování klíčů naimplementovaná a k dispozici.

Dalším výstupem dat z aplikací je síťová komunikace protokolem TCP. Zde jsou důležité a citlivé data šifrovány veřejnými klíči obou serverů RSA (2048b), které klient získá po autentizaci certifikátů. Proti změnám jsou data podepisována pomocí RSA a skrz veřejné klíče v certifikátech snadno ověřena.

Export tokenů na čipovou kartu či zpětné načtení probíhá pouze v případě, že karta je připojena přes vlastní načítací zařízení, které zapojeno pomocí USB k počítači, na kterém běží aplikace Klient, a která má pro ni registrované tokeny.

Bezpečnost

Všechny tři aplikace (Klient, PubServer a AuthServer) je možné chránit loginem a heslem uživatelů či správců. Jelikož většina klíčových funkcí, včetně samotné inicializace a registrace tokenů, je uzamčeno a lze je aktivovat až po řádném přihlášení registrovaného uživatele dané aplikace. Nicméně je implicitně nastaven účet *admin* s heslem *admin*. Vlastní účty lze vytvořit a registrovat, pouze v přihlášeném módu. Loginy se ukládají pro přehlednost v otevřeném textu a hesla se ukládají pro zvýšenou bezpečnost jako otisk hašovací fce SHA - 512.

Dalším bezpečnostním opatřením je šifrování tokenů pomocí RSA a jejich uložení do souboru *sifrovanetokeny.txt*. Zatímco šifrování krátkým veřejným klíčem RSA je poměrně rychlou operací, dešifrování je mnohem náročnější, zvláště při dešifrování více tokenů, které jsou navíc sestaveny z vysokých parametrů. Přenos tokenů na čipovou kartu či pouhé zobrazení šifrovaných tokenů (resp. jejich dešifrování) může trvat několik sekund. V implicitním režimu se

všechny klíče RSA ukládají do klíčového kontejneru OS. Využití možnosti exportování klíčů do textového souboru v otevřené formě je riziko, které uživatel musí sám uvážit.

4.6 Efektivita inicializace a registrování tokenů

Tato část popisuje zejména měření doby inicializace a registrace tokenů u implementovaného systému AASR, respektive jeho aplikací Klient, PubServer a AuthServer.

Výsledky jsou ovlivněny aktuálním zatížením operační paměti, síťovým provozem, atd. Proto měřené charakteristiky jsou značně subjektivní a jejich hodnoty mají pouze orientační informační charakter.

4.6.1 Doba registrace a inicializace tokenů na LocalHostu

Aplikace byly testovány na sestavě s parametry: procesor - Intel Core 2 Duo CPU T6500 @2.10GHz 2.10 GHz, operační paměť - 4 GB, OS - 64 bitový Windows 7 Professional, grafická karta - GeForce G102M CUDA 512 MB. Další software: MS OFFICE ACCESS 2007, utilita Makecert z balíčku MS WINDOWS SDKs, MS Visual Studio 2008.

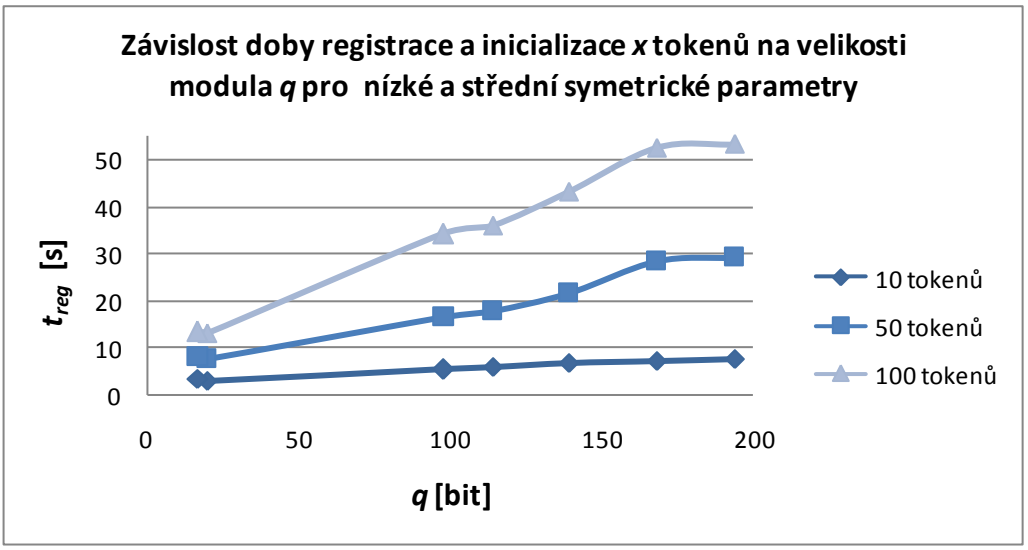
Měření symetrických parametrů

V Tab. 3 jsou zobrazeny průměrné hodnoty celkové doby inicializace a registrace určitého počtu tokenů při malých, středních a vysokých hodnotách symetrických parametrů p a q. Při vyšších parametrech roste i doba celkové inicializace a registrace tokenu. Operace dešifrování a podepisování v RSA má větší časovou náročnost než samotné výpočetní operace umocňovaní a násobení vysokých čísel v modulu p. Při vyšších parametrech roste i délka řetězce vstupního stringu pro operace RSA a roste tak celková doba inicializace a registrace tokenu.

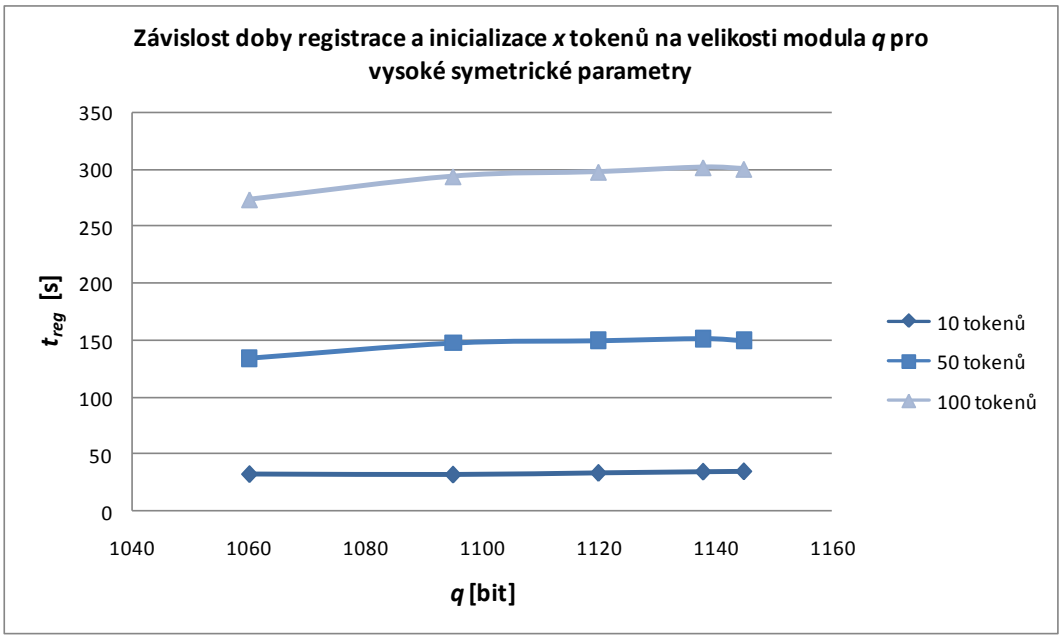
Tab. 3: Doba registrace a inicializace x tokenů při symetrických vysokých, středních a nízkých parametrech kryptosystému AASR (p a q jsou symetrické).

Parametry kryptosystému AASR				Doba trvání t_{reg} registrace a inicializace tokenu na ose AS - K -PA pro počet tokenů pt:				
Délka modulu v bitech		Hodnota generátoru		pt=1	pt=10	pt=50	pt=100	Průměrná doba pro 1 token při pt=10; pt=50; pt=100
p[bit]	q[bit]	g_1[-]	g_2[-]	t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{regpd} [s]
1147	1145	5	16	9,3	34,2	149,2,	300,7	3,4; 3,0; 3,0
1143	1138	34	61	9,7	33,9	151,2	302,1	3,4; 3,0; 3,0
1122	1120	5	16	10	32,9	149,3	298,1	3,3; 3,0; 3,0
1106	1095	277	2445	9,6	31,5	147,2	294,2	3,1; 2,9; 2,9
1064	1060	7	18	9,3	31,9	133,6	274,1	3,2; 2,7; 2,7
201	194	60	157	3,3	7,7	29,3	53,5	0,8; 0,6; 0,5
176	168	153	156	2,9	7,2	28,5	52,8	0,7; 0,6; 0,5
150	139	1031	3776	2,9	6,8	21,6	43,2	0,7; 0,4; 0,4
120	114	258	283	2,8	5,8	17,8	36,0	0,6; 0,4; 0,4
103	98	140	219	2,4	5,3	16,4	34,2	0,5; 0,3; 0,3
27	20	541	866	2	2,9	7,7	13,2	0,3; 0,2; 0,1
27	17	574	731	2,2	3,3	7,9	13,3	0.3; 0,2; 0,1

V Tab. 3. si lze povšimnout rozdílné doby registrace a inicializace tokenu a průměrných dob jednoho tokenu při registraci a inicializaci více tokenů najednou. Tento rozdíl udává všechny operace uvnitř inicializační fáze, které probíhají i při libovolném počtu tokenů pouze jednou. Zejména jde o celou část komunikace na trase Klient - AS, dále certifikaci a ověřovaní na trase Klient - PA a konečného zobrazení a uložení hodnot. Za bezpečné parametry považujeme vysoké symetrické parametry, pro které připadá průměr registrace a inicializace jednoho tokenu přibližně 3 s. Obr. 18 znázorňuje doby registrace 10, 50 a 100 tokenů, které postupně narůstají v závislosti na velikosti prvočíselného parametru q. Systém AASR je bezpečný při velikosti symetrických parametrů p a q nad 1000 bitů, viz Obr. 19. Nízké a střední symetrické parametry zobrazené na Obr. 18 nepovažujeme za bezpečné.



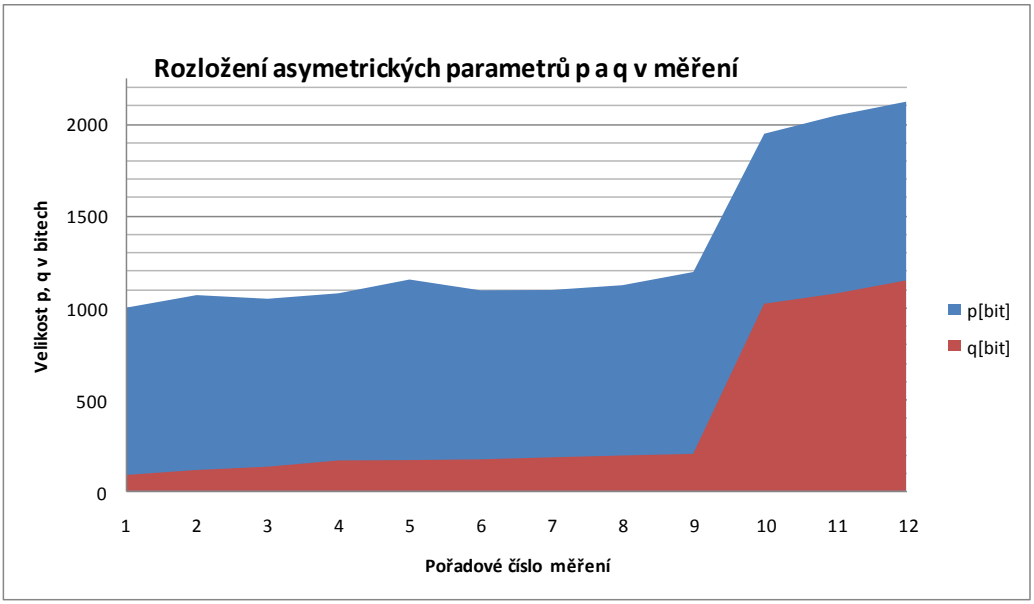
Obr. 18: Závislost doby registrace a inicializace x tokenů na velikosti q pro nízké a střední symetrické parametry.



Obr. 19: Závislost doby registrace a inicializace x tokenů na velikosti q pro vysoké symetrické parametry.

Měření asymetrických parametrů

V Tab. 4 jsou zobrazeny průměrné hodnoty celkové doby inicializace a registrace určitého počtu tokenů při středních a vysokých hodnotách asymetrických parametrů p a q. Při vyšších parametrech roste doba registrace a inicializace tokenu. Střední asymetrické parametry, které považujeme za bezpečné pro $p > 2^{1024}$ a $q > 2^{150}$, mají průměrnou dobu registraci a inicializaci tokenu kolem 1,7 s. V tabulce jsou bezpečné hodnoty podbarveny zeleně. Na Obr. 20 jsou znázorněny odstupy parametrů p a q v bitech. Parametry od 4. měření považujeme za bezpečné.

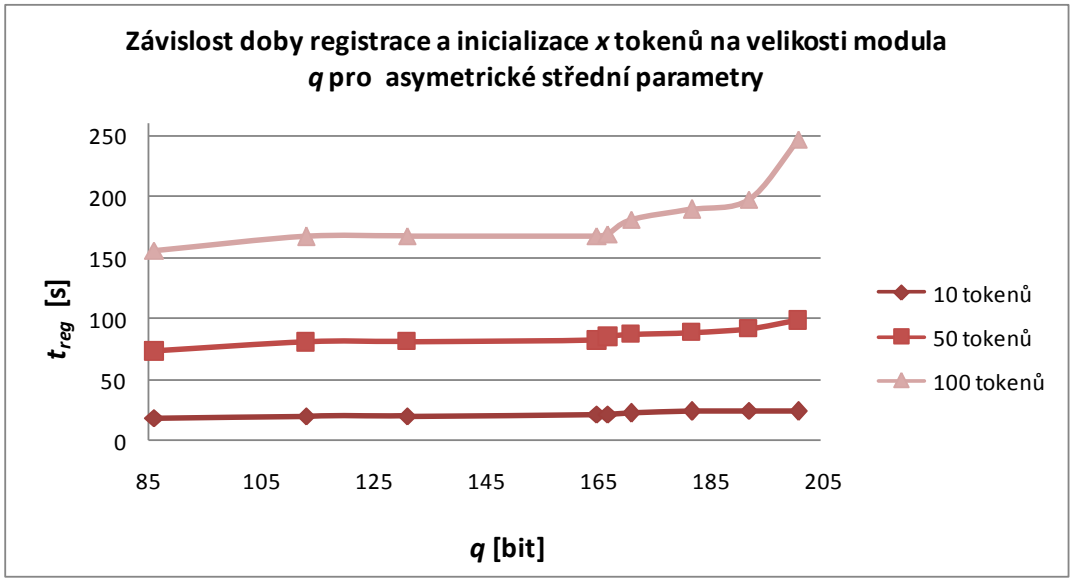


Obr. 20: Rozložení asymetrických parametrů p a q v bitech pro 12 měření.

Tab. 4: Doba registrace a inicializace x tokenů při asymetrických středních a vysokých parametrech kryptosystému AASR (p a q jsou asymetrické, generátory jsou prvky $\mathbb{Z}_p$*).

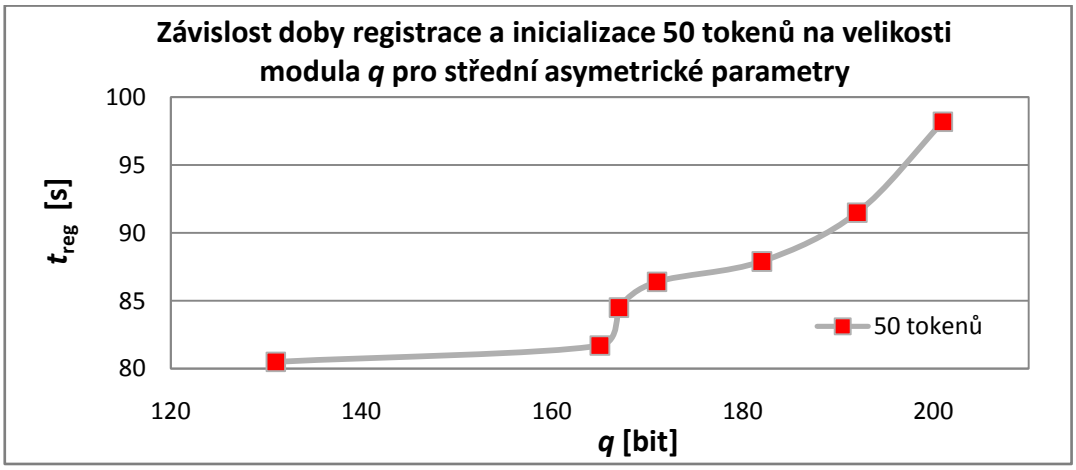
Parametry kryptosystému AASR				Doba trvání t_{reg} registrace a inicializace tokenu na ose AS - K -PA pro počet tokenů pt:				
Délka modulu v bitech		Délka generátoru v bitech		pt=1	pt=10	pt=50	pt=100	Průměrná doba pro 1 token při pt=10; pt=50; pt=100
p[bit]	q[bit]	g_1[bit]	g_2[bit]	t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{regpd} [s]
998	86	988	984	7,2	18,2	72,5	156,1	1,8; 1,5; 1,6
1067	113	1067	1067	7,5	19,6	80,2	167,1	2,0; 1,6; 1,7
1047	131	1045	1046	7,6	19,7	80,5	167,5	2,0; 1,6; 1,7
1077	165	1075	1076	7,8	20,3	81,7	168,1	2,0; 1,6; 1,7
1152	167	1151	1151	7,8	20,7	84,5	169,2	2,1; 1,7; 1,7
1093	171	1091	1093	8,7	21,7	86,4	181,2	2,2; 1,7; 1,8
1095	182	1092	1092	9,1	23,0	87,9	189,4	2,3; 1,8; 1,9
1121	192	1920	1921	9,6	23,4	91,5	198,1	2,3; 1,8; 2,0
1193	201	1193	1193	9,8	23,1	98,2	247,7	2,3; 2,0; 2,5
1947	1019	1946	1947	14,7	45,9	207,2	464,3	4,6; 4,1; 4,6
2045	1074	2043	2043	15,4	48,0	209,8	470,6	4,8; 4,2; 4,7
2122	1146	2120	2121	17,2	62,9	226,5	509,2	6,3; 4,5; 5,1

Pokud tedy porovnáme doby registrace a inicializace jednoho tokenu u bezpečných symetrických parametrů (2,9 s) a asymetrických bezpečných parametrů (1,7 s), zjistíme, že použití asymetrických parametrů *p* a *q* je efektivnější a navíc přináší méně náročné hodnoty pro početní operace na čipové kartě. Na Obr. 21 je znázorněna závislost doby registrace a inicializace *x* tokenů na velikosti *q* pro asymetrické parametry. Průběhy mají lehce nelineární charakter. Příčinou odchylek je aktuální vytížení procesoru a operační paměti na testovaném počítači.



Obr. 21: Závislost doby registrace a inicializace *x* tokenů na velikosti *q* pro asymetrické střední parametry.

Na Obr. 21 je zobrazena závislost doby registrace a inicializace 10, 50 a 100 tokenů na velikosti *q* pro asymetrické střední parametry. Kolem hodnoty $q = 190$ bitů začne doba registrace rychleji narůstat. Tato skutečnost může být způsobena i zvýšením doby podpisu RSA, které roste nelineárně na délce zprávy respektive tokenu. To je způsobeno dlouhým soukromým podpisovým klíčem RSA. Na Obr. 22 je zobrazena závislost doby registrace a inicializace 50 tokenů na velikosti modula *q* při asymetrických středních parametrech. Detailní pohled potvrzuje nelineární charakter růstu doby registrace tokenu při vyšších parametrech *q* a *p*.



Obr. 22: Závislost doby registrace a inicializace 50 tokenů na velikosti *q* pro střední asymetrické parametry.

4.6.2 Doba registrace a inicializace tokenů v místní síti

Měření doby registrace a inicializace tokenů v místní síti rovněž slouží pouze k orientační představě funkčnosti aplikace. V reálném provozu v rozsáhlé síti bude hrát velkou roli zpoždění a propustnost linek či celkové cesty klienta k serverům AS a PA.

Testování proběhlo pouze na počítačích v místní síti, které jsou připojeny pomocí kabeláže UTP (enhanced category 5) k jednomu směrovači (D-link DSL-G684T). Délka kabeláže od počítače k směrovači je cca 5-10 metrů. Rychlost každé linky je 100 Mb/s. Celková doba registrace a inicializace tokenů je ovlivněna také výpočetním výkonem jednotlivých stanic. Výpočetní výkony jsou porovnány v Tab. 5, kde měření bylo provedeno pro stejné asymetrické střední parametry na místní smyčce (LocalHost). U počítače PC3, který disponuje nejslabším výkonem, docházelo často k potížím při náročném dešifrování RSA u aplikace PA. Proto doporučuji testovat aplikaci na počítači s vyšším výkonem a nejlépe na OS Windows 7, kde byla aplikace nejvíce otestována. U silnějšího počítače PC2 byla doba registrace a inicializace tokenů nejkratší (69,7 s pro registraci a inicializaci 50 tokenů). Na počítači PC2 běží operační systém Windows XP. Na rozdíl od PC3, u testování aplikací na PC2 nedocházelo k žádným chybám. Aplikace na PC2 a PC3 používaly pouze vlastní certifikaci. U registrování většího počtu tokenů například 100 tokenů dochází k občasným komunikačním chybám, které způsobují dlouhé stringy dat, které aplikace musí dešifrovat a zpracovávat.

PC1:
procesor - Intel Core 2 Duo CPU T6500 @2.10GHz 2.10 GHz, operační paměť - 4 GB, OS - 64 bitový Windows 7 Professional, grafická karta - GeForce G102M CUDA 512 MB.

PC2:
procesor - AMD Athlon(tm) 64 X2 DUAL Core Processor 4200+ 2.21GHz 2.10 GHz, operační paměť - 1 GB, OS - 32 bitový MS Windows XP Professional.

PC3:
procesor - Intel(R) Celeron(R) M CPU 420 @ 1,60 GHz, operační paměť - 896MB RAM, OS - 32 MS Windows XP Professional.

Tab. 5: Srovnání výkonu na měřených PC: Doba registrace a inicializace x tokenů při asymetrických středních parametrech kryptosystému AASR (p a q jsou asymetrické, generátory jsou prvky $\mathbb{Z}_p$*).

Parametry kryptosystému AASR				Počítač	Doba trvání t_{reg} registrace a inicializace tokenu na ose AS - K - PA pro počet tokenů pt:				
Délka modulu v bitech		Délka generátoru v bitech			pt=1	pt=10	pt=50	pt=100	Průměrná doba pro 1 token při pt=10; pt=50; pt=100
p[bit]	q[bit]	g_1[bit]	g_2[bit]		t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{regpd} [s]
1112	163	1112	1111	PC1	8,1	21,1	86,5	171,1	2,1; 1,7; 1,7
				PC2	7,5	17,3	69,7	151,3	1,7; 1,4; 1,5
				PC3	10,8	25,4	112,3	223,5	2,5; 2,2; 2,2

V Tab. 6 jsou změřeny doby registrace a inicializace tokenů při asymetrických středních parametrech na aplikacích, které jsou rozmístěny v místní síti. Aplikace klient (ProjektAASRv1) běží na PC1, aplikace PubServer běží na PC2 a aplikace AuthServer běží na PC3. Všechny aplikace měly nastaveny vlastní certifikaci.

Tab. 6: Měření v místní síti: Doba registrace a inicializace *x* tokenů při asymetrických středních parametrech kryptosystému AASR (*p* a *q* jsou asymetrické, generátory jsou prvky $\mathbb{Z}_p^*$).

Parametry kryptosystému AASR				Doba trvání t_{reg} registrace a inicializace tokenu na ose AS - K - PA pro počet tokenů pt:				
Délka modulu v bitech		Délka generátoru v bitech		pt=1	pt=10	pt=20	pt=50	Průměrná doba pro 1 token při pt=10; pt=20; pt=50
p[bit]	*q*[bit]	g_1[bit]	g_2[bit]	t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{reg} [s]	t_{regpd} [s]
1093	101	1091	1091	9,1	24,3	39,2	90,7	2,7; 2,0; 1,8
1109	170	1109	1109	9,9	23,3	39,4	96,7	2,3; 2,0; 1,9

4.7 Shrnutí

Implementace systému AASR v prostředí .NET v jazyce *C#* je ve funkčním stavu a k dispozici pro další testování a případnou další optimalizaci dílčích funkcí. Implementace se skládá ze tří aplikací Klient (respektive ProjektAASRv1), PubServer a AuthServer. Každá aplikace plní svoje příslušné funkce a své části protokolu inicializace a registrace tokenu, tak jak popisuje část 6.1. Aplikace jsou implicitně přednastaveny na bezpečnější chování a funkce. Například u aplikace Klient není po spuštění povolen zápis tokenů do souboru *tokeny.xml*, či u PubServer je rovněž vypnut zápis registrovaných tokenů do *tokeny.xml*. Implicitně je také přednastavena certifikace X.509 a u všech operací RSA se implicitně používá klíčový kontejner OS. Povolení exportu klíčů RSA do otevřeného textového souboru představuje bezpečnostní riziko a stejně jako použití vlastní certifikace má pouze testovací účel.

Protokol registrace a inicializace tokenů byl první testován na LocalHostu. Doba registrace a inicializace jednoho tokenu pro střední asymetrické parametry *p* a *q* se pohybuje kolem 2 sekund. Při větším počtu registrovaných tokenů (přes 50 tokenů) klesá délka registrace jednoho tokenu na hodnotu 1,7 s. Například celková doba registrace a inicializace 50 tokenů pro vybrané asymetrické parametry (délka *p* = 1077 bitů a *q* = 165 bitů) na PC1 trvala celkem 81,7 s a průměrná doba registrace a inicializace jednoho tokenu byla 1,6 s. Asymetrické střední parametry *p* a *q* jsou o něco efektivnější, jak pro inicializaci a registraci tokenu, tak pro početní operace na čipové kartě (smart card), než symetrické vysoké parametry poskytující srovnatelnou bezpečnost. Aplikace byly testovány i v místní síti. Testy byly provedeny pro nastavení s vlastní certifikací a doba registrace jednoho tokenu byla přibližně o 1 s delší než na LocalHostu.

Pro spuštění a otestovaní aplikací je na přiloženém DVD návod pro instalaci a nastavení aplikací (ManualInstalace.txt). Testování aplikací je doporučeno na MS Windows XP a Windows 7. Slabší sestavy než uvedené PC1 a PC2 můžou vykazovat problémy s dešifrováním tokenů. Stabilnější a rychlejší jsou registrace menšího počtu tokenů (pod 20 tokenů) a registrace při nižších parametrech *p* a *q* (střední asymetrické parametry). Při exportu tokenů na čipovou kartu je třeba respektovat omezení paměti čipové karty. Testovaná čipová karta disponovala volnou pamětí cca 30 KB, kterou z velké části využije pro výpočty. Počet tokenů, které doprovází i podpis serveru PA, je pamětí čipové karty limitován na cca 3 tokeny při středních asymetrických parametrech. Další informace o čipové kartě v [49].

Aplikace Klient po úspěšné registraci a inicializaci tokenů může bezpečně předávat tokeny pomocí čtecího zařízení dále na čipovou kartu, která tokeny dále použije k anonymní autentizaci v systému AASR.

ZÁVĚR

Ochranou soukromí na Internetu se zabývají především systémy a protokoly anonymní autentizace. Systémy anonymní autentizace můžou být realizovány pomocí vhodných protokolů a schémat, které byly uvedeny v první kapitole. Jedná se především o identifikaci na konceptu nulové znalosti, závazkové schéma, nepopiratelné podpisy a slepé podpisy, které jsou obvykle založeny na kryptografických primitivech RSA, DL, hašovací funkce, atd.

Jedním ze systémů, které vyžadují anonymní autentizaci, je systém elektronické hotovosti (e-cash), který je popsán a analyzován ve druhé kapitole. Doposud navrhované koncepty e-cash byly bezpečné v modelu RO, při nejlepší dosažené efektivitě $O(l+k)$, kde l je počet el. mincí a k je bezpečnostní parametr, viz [25] či [30]. Do budoucna je ale potřeba přejít na standardní bezpečnostní model (reálný model) a zlepšit efektivitu výpočetních operací. Proti praktické implementaci e-cash stojí také malý zájem o anonymitu ze strany uživatelů, kterým prozatím vyhovují osvědčené praktiky online nákupů a služeb, což také vyhovuje i poskytovatelům služeb, obchodům a bankám.

Skupinové podpisy se v podstatě snaží docílit těchto vlastností: bezpečnost ve standardním modelu, efektivita klíčových algoritmů (zápis, podpis, ověření, trasování), krátký podpis a vhodná konstrukce pro danou aplikaci či systém. Mezi návrhy schémat skupinových podpisů, které tyto vlastnosti téměř splňují, lze zařadit BW07 [16] a schéma G07 [42], které jsou obě bezpečné ve standardním modelu, používají bilineární grupy a jejich podpis je konstantní v závislosti na velikosti skupiny. G07 [42] navíc umožňuje dynamické připojení nových členů. Objevila se i další schémata, která se snažila vylepšovat dílčí části a zefektivňovat jednotlivé algoritmy. Jako například dávkové ověřovaní, multi-připojování a lokální verifikaci, atd.

Výsledky analýzy anonymní autentizace poukazují na velký počet návrhů schémat se skupinovými podpisy, které jsou popsány v kapitole 2 a jejich vlastnosti shrnuty v Tab. 1. Rovněž se objevuje i poměrně velký počet návrhů systému el. hotovosti, které však nedosahují takových kvalit jako schémata skupinových podpisů. Na rozdíl od pokročilých schémat skupinových podpisů postrádají bezpečnost ve standardním modelu a mají problém s efektivitou při velkém počtu el. mincí. Větší uplatnění nacházejí anonymní autentizační systémy AAS, které poskytují přístup k chráněným službám výměnou za tokeny. V kapitole 3 jsou rozebrány základní typy AAS, přičemž nejspravedlivější z pohledu klienta se jeví schémata, kde je funkce registrace, inicializace a odhalení identity rozdělena na dvě nebo více entit. Objevují se pokročilé návrhy pro k-násobné použití tokenů, jejich delegovatelnost a také první implementace, např. systém Idemix v jazyce Java.

Praktická část, která je popsána v kapitole 4, představuje implementaci (v prostředí .NET v jazyce C#) systému AAS. Tento systém je vyvíjen na fakultě FEKT VUTBR, viz [43]. Inicializace tokenů a odhalení identity je rozděleno mezi server veřejné autority a poskytovatele služeb (autentizační server). Implementace přináší tři funkční aplikace: ProjektAASRv1 (klient), AuthServer (autentizační server) a PubServer (server veřejné autority), které spolu komunikují a zajišťují registraci klientů, inicializaci a registraci tokenů a jiné doplňující funkce. V aplikaci AuthServer lze vygenerovat parametry kryptosystému AASR, provést registraci nových uživatelů a inicializovat tokeny. Aplikace PubServer inicializuje tokeny a předává je zpět klientské aplikaci ProjektAASRv1. Aplikace ProjektAASRv1 může najednou inicializovat a registrovat až 100 tokenů. Aplikace byly podrobně otestovány a byly změřeny orientační doby registrace a inicializace různého počtu tokenů při bezpečných parametrech p a q. Průměrná hodnota inicializace a registrace jednoho tokenu se pohybuje kolem 1,7 s. Aplikace byly otestovány na místní síti, kde průměrná doba inicializace a registrace jednoho tokenu se pohybuje kolem 2,0 s. Podepsané tokeny lze bezpečně exportovat na čipovou kartu, kterou se můžeme poté anonymně autentizovat u určitého přístupového bodu.

LITERATURA

[1] ATENIESE, Giuseppe, CAMENISCH, Jan, HOHENBERGER, Susan, MEDEIROS, Breno de. Practical Group Signatures without Random Oracles. *EUROCRYPT'06* [online]. 2006 [cit. 2009-11-20].

[2] ATENIESE, Giuseppe, CAMENISCH, Jan, JOYE, Marc, TSUDIK, Gene. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. [online]. 2000 [cit. 2009-11-20].

[3] AU, Man Ho, SUSILO, Willy, MU, Yi. Constant-Size Dynamic k-TAA. *SCN, volume 4116 of LNCS* [online]. 2006 [cit. 2010-3-20].

[4] AU, Man Ho, KAPADIA, Apu, SMITH, Sean, TSANG, Patrick. BLAC: Revoking Repeatedly Misbehaving Anonymous Users Without Relying on TTPs. [online]. 2008 [cit. 2010-3-20].

[5] BACKES, Michael, CAMENISCH, Jan, SOMMER, Dieter. Anonymous yet Accountable Access Control. *Proceedings of the 2005 ACM Workshop on Privacy in The Electronic Society* [online]. 2005 [cit. 2009-11-10].

[6] BELENKIY, Mira, CHASE, Melissa, KOHLWEISS, Markulf, LYSYANSKAYA, Anna. Non-Interactive Anonymous Credentials. *IACR Cryptology* [online]. 2007 [cit. 2010-3-4].

[7] BELENKIY, Mira, CAMENISCH, Jan, CHASE, Melissa, KOHLWEISS, Markulf, LYSYANSKAYA, Anna, SHACHAM, Hovav. Randomizable Proofs and Delegatable Anonymous Credentials. *Proceedings of the 29th Annual International Cryptology Conference on Advances in Cryptology* [online]. 2009 [cit. 2009-11-14].

[8] BELLARE, Mihir, MICCIANCIO, Daniele, WARINSCHI, Bogdan. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. *Advances in Cryptology—EUROCRYPT 2003* [online]. 2003 [cit. 2009-11-24]. Dostupný z WWW: <http://www.iacr.org/archive/eurocrypt2003/26560615/26560615.pdf>.

[9] BELLARE, Mihir, ROGAWAY, Phillip. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. *Proceedings of the 1st ACM Conference on Computer and Communications Security (1993)* [online]. 1993 [cit. 2009-11-14].

[10] BELLARE, Mihir, SHI, Haixia, ZHANG, Chong. Foundations of Group Signatures: The Case of Dynamic Groups. *CT-RSA* [online]. 2005 [cit. 2009-11-10].

[11] BENJUMEA, Vicente, CHOI, Seung Geol, LOPEZ, Javier, YUNG, Moti. Fair Traceable Multi-Group Signatures. *LNCS 2008* [online]. 2008 [cit. 2009-11-10].

[12] BONEH, Dan, BOYEN, Xavier. Short signatures without random oracles. *LNCS 2004* [online]. 2004 [cit. 2009-11-20].

[13] BONEH, Dan, BOYEN, Xavier, SHACHAM, Hovav. Short Group Signatures. *Advances in Cryptology–CRYPTO '04* [online]. 2004 [cit. 2009-11-10].

[14] BONEH, Dan, SHACHAM, Hovav. Group Signatures with Verifier-Local Revocation. [online]. 2004 [cit. 2009-11-10].

[15] BOYEN, Xavier, WATERS, Brent. Compact Group Signatures Without Random Oracles. [online]. 2006 [cit. 2009-11-20].

[16] BOYEN, Xavier, WATERS, Brent. Full-Domain Subgroup Hiding and Constant-Size Group Signatures. [online]. 2007 [cit. 2009-11-20].

[17] BRANDS, Stefan. Offline electronic cash based on secret-key certificates. World Wide Web electronic publication, 2009.

[18] BRANDS, Stefan. Electronic cash. World Wide Web electronic publication, 2009.

[19] BRICKELL, Ernie, CAMENISCH, Jan, CHEN, Liqun. Direct Anonymous Attestation. *CCS´04* [online]. 2004 [cit. 2010-3-20].

[20] BURDA, Karel. *Bezpečnost informačních systémů.* Brno : [s.n.], 2005. 104 s.

[21] CAMENISCH, Jan. Efficient Anonymous Fingerprinting with Group Signatures. *Lecture notes in computer science* [online]. 2000 [cit. 2009-11-10].

[22] CAMENISCH, Jan, DAMGARD, Ivan. Verifiable Encryption, Group Encryption, and Their Applications to Separable Group Signatures and Signature Sharing Schemes. *Lecture notes in computer science* [online]. 2000 [cit. 2009-11-10].

[23] CAMENISCH, Jan, GROTH, Jens. Group Signatures: Better Efficiency and New Theoretical Aspects. *SCN 2004* [online]. 2004 [cit. 2009-11-10].

[24] CAMENISCH, Jan, HOHENBERGER, Susan, KOHLWEISS, Markulf, LYSYANSKAYA, Anna, MEYEROVICH, Mira. How to win clonewars: efficient periodic n-times anonymous authentication. *ACM* [online]. 2006 [cit. 2010-1-04].

[25] CAMENISCH, Jan, HOHENBERGER, Susan, LYSYANSKAYA, Anna. Compact E-Cash. *Advances in Cryptology–EUROCRYPT 2005* [online]. 2006 [cit. 2009-11-04].

[26] CAMENISCH, Jan, HOHENBERGER, Susan, LYSYANSKAYA, Anna. Balancing Accountability and Privacy Using E-Cash (Extended Abstract). [s.l.] : [s.n.], 2006. ISBN 978-3-540-380. s. 141-155.

[27] CAMENISCH, Jan, KOPROWSKI, Maciej, WARINSCHI, Bodgan. Efficient Blind Signatures without Random Oracles. *SCN 2004* [online]. 2004 [cit. 2009-11-10].

[28] CAMENISCH, Jan, LYSYANSKAYA, Anna. Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation. *EUROCRYPT 2001* [online]. 2001 [cit. 2009-11-27].

[29] CAMENISCH, Jan, LYSYANSKAYA, Anna. Signature Schemes and Anonymous Credentials from Bilinear Maps. *Lecture notes in computer science* [online]. 2004 [cit. 2009-11-10].

[30] CAMENISCH, Jan, MEYEROVICH, Mira, LYSYANSKAYA, Anna. Endorsed E-Cash. *Proceedings of the 2007 IEEE Symposium on Security and Privacy* [online]. 2007 [cit. 2009-11-04], s. 101-115. ISSN 0-7695-2848-1.

[31] CAMENISCH, Jan, MICHELS, Markus. A Group Signature Scheme Based on an RSA-Variant. [online]. 1998 [cit. 2009-11-10].

[32] CAMENISCH, Jan, STADLER, Markus. Efficient Group Signature Schemes for Large Groups. [online]. 1997 [cit. 2009-11-10].

[33] CAMENISCH, Jan, VAN HERREWEGHEN, Els. Design and Implementation of the idemix Anonymous Credential System. *Proceedings of the 9th ACM Conference on Computer and Communications Security (2002)* [online]. 2002 [cit. 2009-11-27].

[34] CRAMER, Donald, DAMGARD, Ivan, NIELSEN, Jesper Buus. On Electronic Payment Systems. *CPT 2009* [online]. 2009 [cit. 2009-11-04]. Dostupný z WWW: <http://www.daimi.au.dk/~ivan/ecash.pdf>.

[35] DAMGARD, Ivan, NIELSEN, Jesper Buus. Commitment Schemes and Zero-Knowledge Protocols. [online]. 2008 [cit. 2009-11-10]. Dostupný z WWW: <http://www.daimi.au.dk/~ivan/ComZK06.pdf>.

[36] DING, Xuhua, TSUDIK, Gene, XU, Shouhuai. Leak-free Mediated Group Signatures. *Journal of Computer Security, 2009* [online]. 2009 [cit. 2009-11-20]. Dostupný z WWW: <http://www.mysmu.edu/faculty/xhding/publications/jmgs.pdf>.

[37] DOSTÁLEK, Libor, VOHNOUTOVÁ, Marta. *Velký průvodce infrastrukturou PKI a technologií elektronického podpisu* . [s.l.] : Computer Press, a. s., 2006. 534 s. ISBN 80-251-0828-7.

[38] FERRARA, Anna Lisa, GREEN, Matthew, HOHENBERGER, Susan, PEDERSEN, Michael. Practical Short Signature Batch Verification. *Topics in Cryptology-CT-RSA 2009* [online]. 2009 [cit. 2009-11-10].

[39] FUJISAKI, Eiichiro, OKAMOTO, Tatsuaki. Statistical zero knowledge protocols to prove modular polynomial relations. *Advances in Cryptology–CRYPTO '97* [online]. 1997 [cit. 2009-11-10].

[40] FURUKAWA, Jun, IMAI, Hideki. An Efficient Group Signature Scheme from Bilinear Maps. [online]. 2006 [cit. 2009-11-20].

[41] GOLDBERG, Ian. On the Security of the Tor Authentication Protocol. *Lecture Notes in Computer Science* [online]. 2006 [cit. 2009-11-20].

[42] GROTH, Jens. Fully Anonymous Group Signatures without Random Oracles. *Advances in Cryptology - ASIACRYPT 2007* [online]. 2007 [cit. 2009-11-10].

[43] HAJNÝ, Jan. Anonymous Authentication for Smartcards. *Radioengineering*. 2010, -, s. 1-7. ISSN 1210-2512.

[44] HELBACH, Jörg, SCHWENK, Jörg, SCHÄGE, Sven. Code Voting With Linkable Group Signatures. *E-voting.cc 2008* [online]. 2008 [cit. 2009-11-10].

[45] CHAUM, David, FIAT, Amos, NAOR, Mori. Untraceable electronic cash. *Advances in Cryptology — CRYPTO '88* [online]. 1988 [cit. 2009-11-04].

[46] CHAUM, David, VAN HEYST, Eugene. Group Signatures. *Advances in Cryptology – EUROCRYPT '91* [online]. 1991 [cit. 2009-11-10].

[47] CHEN, L., PEDERSEN, T.P. New Group Signature Schemes. *Advances in Cryptology – EUROCRYPT '94* [online]. 1995 [cit. 2009-11-10].

[48] IBRAHIM, Maged Hamada. Resisting Traitors in Linkable Democratic Group Signatures. *International Journal of Network Security* [online]. 2009 [cit. 2009-11-10].

[49] JURAS, S. *Autentizace pomocí smartkaret*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2010. Vedoucí diplomové práce Ing. Jan Hajný.

[50] KIAYIAS, Aggelos, YUNG, Moti. Group Signatures: Provable Security, Efficient Constructions and Anonymity from Trapdoor-Holders. *IACR e-print, 2004* [online]. 2004 [cit. 2009-11-10]. Dostupný z WWW: <http://eprint.iacr.org/2004/076.pdf>.

[51] KIAYIAS, Aggelos, YUNG, Moti. Group Signatures with Efficient Concurrent Join. *Advances in Cryptology – EUROCRYPT '05* [online]. 2005 [cit. 2009-11-10].

[52] KIAYIAS, Aggelos, ZHOU, Hong-Sheng. Hidden Identity-Based Signatures. [online]. 2008 [cit. 2009-11-20]. Dostupný z WWW: <http://eprint.iacr.org/2007/140.pdf>.

[53] LEE, Cheng-Chi, CHANG, Ting-Yi, HWANG, Min-Shiang. A New Group Signature Scheme Based on the Discrete Logarithm. *Journal of Information Assurance and Security 2010* [online]. 2009 [cit. 2009-11-10].

[54] LEI, Chaoqin, KOU, Weidong, FAN, Kai, FAN, Wei, An Traceable Electronic Cash Model Based on Group Signature. World Wide Web electronic publication, 2009.

[55] LI, Xiang-xue, QIAN, Hai-feng, LI, Jian-hua. Democratic Group Signatures with Threshold Traceability. [online]. 2009 [cit. 2009-11-10].

[56] LIPMAA, Helger, LAUR, Sven. A New Protocol for Conditional Disclosure of Secrets And Its Applications. *ACNS 2007* [online]. 2007 [cit. 2009-11-10].

[57] LYSYANSKAYA, Anna, RAMZAN, Zulfikar. Blind Digital Signatures: A Scalable Solution to Electronic Cash. [online]. 1998 [cit. 2009-11-10].

[58] MENEZES, Alfred, VAN OORSCHOT, Paul, VANSTONE, Scott. *Handbook of applied cryptography.* Boca Raton : CRC Press, 1997. 780 s. ISBN 0849385237.

[59] NGUYEN, Lan, SAFAVI-NAINI, Rei. Dynamic k-times anonymous authentication. *ACNS* [online]. 2005 [cit. 2010-28-3].

[60] NGUYEN, Lan, SAFAVI-NAINI, Rei. Efficient and Provably Secure Trapdoor-free Group Signature Schemes from Bilinear Pairings. [online]. 2004 [cit. 2009-11-20].

[61] PEDERSEN, Torben Pryds. Non-interactive and Information- theoretic secure verifiable secret sparing. *CRYPTO '91* [online]. 1991 [cit. 2009-11-10].

[62] POPESCU, Konstantin. An Efficient ID-Based Group Signature Scheme. [online]. 2002 [cit. 2009-11-20].

[63] SANDER, Tomas, TA-SHMA, Ammon. Auditable, Anonymous Electronic Cash Extended Abstract. *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology* [online]. 1999 [cit. 2009-11-04], s. 552-572.

[64] SCHNORR, C.: Efficient signature generation by smart cards. *Journal of Cryptology*. [online]. 1991 [cit. 2009-11-10].

[65] SCHWARTZ, Edward, BRUMLEY, David, McCUNE, Jonathan. Contractual Anonymity. [online]. 2009 [cit. 2010-3-20].

[66] SHAHANDASHTI, Siamak F, SAFAVI-NAINI, Reihaneh. Threshold Attribute-Based Signatures and Their Application to Anonymous Credential Systems. [online]. 2009 [cit. 2009-11-10]. Dostupný z WWW: <http://eprint.iacr.org/2009/126.pdf>.

[67] SHACHAM, Hovav, WATERS, Brent. Efficient Ring Signatures Without Random Oracles. *Lecture notes in computer science* [online]. 2007 [cit. 2009-11-10].

[68] STALLINGS, William. *Cryptography and Network Security: Principles and Practises.* 4th edition. Upper Saddle River : Prentice Hall, 2006. 680 s. ISBN 0131873164, 97801.

[69] TERANISHI, Isamu, FURUKAWA, Jun, SAKO, Kazue. k-times anonymous authentication (extended abstract). *ASIACRYPT 2004* [online]. 2004 [cit. 2010-3-28].

[70] TSANG, Patrick, SMITH, Sean. PPAA: Peer-to-Peer Anonymous Authentication. [online]. 2008 [cit. 2010-3-28].

[71] WEI, Victor K. Tracing-by-Linking Group Signatures. [online]. 2005 [cit. 2009-11-20].

[72] ZHOU, Sujing, LIN, Dongdai. Unlinkable Randomizable Signature and Its Application in Group Signature. [online]. 2008 [cit. 2009-11-20].

[73] ZHU, Bo, SETIA, Sanjeev, JAJODIA, Sushil. Providing Witness Anonymity in Peer-to-Peer Systems. [online]. 2006 [cit. 2010-3-28].

[74] Rozšíření aplikace *Emil.GMP.dll* pro práci s velkými čísly. Dostupný z WWW: http://www.emilstefanov.net/Projects/GnuMpDotNet/

SEZNAM POUŽITÝCH ZKRATEK

AAS	Anonymous Authentication Systems
AASR	Anonymous Authentication with Spread Revelation
AES	Advanced Encryption Standart
CA	Certifikační Autorita
CDH	Computational Diffie-Hellman
CHL	Camenisch Hohenberger Lysyanskaya eCash scheme
DDH	Decision Diffie-Hellman
DH	Diffie-Hellman
DL	Discrete Logarithm
DLDH	Decision Linear Diffie-Hellman
DLP	Discrete Logarithm Problem
DSA	Digital Signature Algorithm
FFS	Fiege-Fiat-Shamir
GQ	Guillou-Quisquater
HSDH	Hidden Strong Diffie-Hellman
HVZK	Honest Verifier Zero Knowledge
IF	Integer Factorization
IS	Informační Systém
MAC	Message Authentication Code
NIPDLE	Non-Interactive Proof of Discrete Logarithm Equivalency
NIZK	Non-Interactive Zero-Knowledge
PA	Public Authority
PDLK	Proof of Discrete Logarithm Knowledge
PGP	Pretty Good Privacy
PK	Public Key
PKI	Public Key Infrastructure
RA	Registrační Autorita
RSA	Rivest, Shamir a Adleman
SDH	Strong Diffie-Hellman
SDP	Subgroup Decision Problem
SHA	Secure Hash Algorithm
SK	Secret Key
SP	Service Provider
SRSA	Strong RSA
TPM	Trusted Platform Module
TTP	Trusted Third Party
XML	eXtensible Markup Language
ZKIP	Zero-Knowledge Interactive Proof system
ZKPK	Zero-Knowledge Proof of Knowledge

SEZNAM PŘÍLOH

Příloha A: Obsah přiloženého DVD

Složka s programy: ProjektAASR\

Manuál k instalaci a pokyny k obsluze aplikací:
ManualInstalace.txt

Spuštění implementace (Solution) v MS Visual Studio:
ProjektAASRv1.sln

Složka aplikace autentizační server:
AuthServer\
spuštění aplikace: AuthServer\bin\Debug\AuthServer.exe

Složka aplikace PA server:
PubServer\
spuštění aplikace: PubServer\bin\Debug\PubServer.exe

Složka aplikace klient:
ProjektAASRv1\
spuštění aplikace: ProjektAASRv1\bin\Debug\ProjektAASRv1.exe

Složka databáze klientů AASR:
ProjektAASR-databaze-ServerAS\

Složka podpůrné třídy Client:
Client\
-neobsahuje spustitelnou aplikaci

podpůrný soubor:
Rijndael.bin

Složka obsahující snímky a videa z testování: Snímky a videa\